Red Hat Ceph Storage 7

文件系统指南

配置和挂载 Ceph 文件系统

Last Updated: 2024-06-14

Red Hat Ceph Storage 7 文件系统指南

配置和挂载 Ceph 文件系统

法律通告

摘要

本指南介绍了如何配置 Ceph 元数据服务器(MDS)以及如何创建、挂载和工作 Ceph 文件系统(CephFS)。 红帽致力于替换我们的代码、文档和 Web 属性中存在问题的语言。我们从这四个术语开始：master、slave、黑名单和白名单。由于此项工作十分艰巨，这些更改将在即将推出的几个发行版本中逐步实施。详情请查看 CTO Chris Wright 信息。

目录

第 1 章 CEPH 文件系统简介

作为存储管理员，您可以了解管理 Ceph 文件系统 (CephFS) 环境的功能、系统组件和限制。

1.1. CEPH 文件系统的功能和增强

Ceph 文件系统 (CephFS) 是兼容 POSIX 标准的文件系统，在 Ceph 的分布式对象存储基础上构建，称为 RADOS（可靠的自主分布式对象存储）。CephFS 提供对 Red Hat Ceph Storage 集群的文件访问，并尽可能使用 POSIX 语义。例如，与 NFS 等其他常见网络文件系统相比，CephFS 在客户端之间保持强大的缓存一致性。目标是让使用文件系统的进程的行为与位于同一主机上的进程在不同的主机上的行为相同。但在某些情况下，CephFS 会偏离严格的 POSIX 语义。

Ceph 文件系统具有以下功能和增强功能：

可扩展性

Ceph 文件系统具有高度可扩展性，因为元数据服务器水平扩展，并且直接客户端对各个 OSD 节点进行读写操作。

共享文件系统

Ceph 文件系统是一种共享文件系统，因此多个客户端可以同时处理同一文件系统。

多文件系统

您可以在一个存储集群中有多个文件系统。每个 CephFS 都有自己的一组池，以及自己的一组元数据服务器(MDS)的排名。在部署多个文件系统时，这需要更多运行 MDS 守护进程。这可提高元数据吞吐量，但会增加操作成本。您还可以限制客户端对特定文件系统的访问。

高可用性

Ceph 文件系统提供 Ceph 元数据服务器 (MDS) 的集群。一个处于活动状态，另一些处于待机模式。如果活动的 MDS 意外终止，其中一个备用 MDS 就会变为活跃状态。因此，客户端挂载会继续处理服务器故障。此行为使 Ceph 文件系统高度可用。另外，您可以配置多个活跃的元数据服务器。

可配置文件和目录

Ceph 文件系统允许用户配置文件和目录布局，以跨对象使用多个池、池命名空间和文件分条模式。

POSIX 访问控制列表 (ACL)

Ceph 文件系统支持 POSIX 访问控制列表 (ACL)。默认启用 ACL，将 Ceph 文件系统挂载为带有内核版本 **kernel-3.10.0-327.18.2.el7** 或更新版本的内核客户端。若要将 ACL 与挂载为 FUSE 客户端的 Ceph 文件系统搭配使用，您必须启用它们。

客户端配额

Ceph 文件系统支持在系统中的任何目录上设置配额。配额可以限制目录层次结构中该点下存储的字节数或文件数量。CephFS 客户端配额默认为启用。

重要

CephFS EC 池仅用于归档目的。

其它资源

- 请参阅 *操作指南中的 Ceph 编排器部分管理 MDS 服务*，以安装 Ceph 元数据服务器。
- *请参阅文件系统指南中的 Ceph 文件系统部署* 一节，以创建 Ceph 文件系统。

1.2. CEPH 文件系统组件

Ceph 文件系统有两个主要组件：

客户端

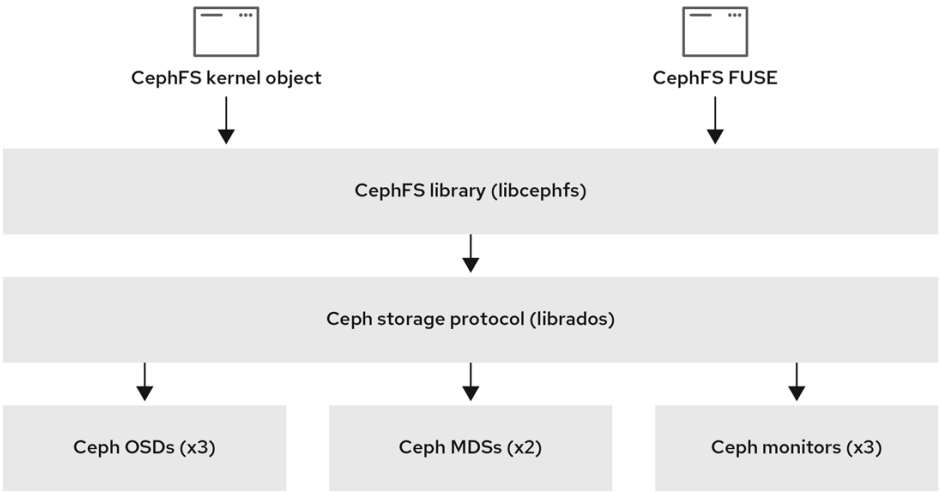
CephFS 客户端代表使用 CephFS 的应用执行 I/O 操作，如用于 FUSE 客户端的 **ceph-fuse**，**kcephfs** 用于内核客户端。CephFS 客户端向活跃的元数据服务器发送元数据请求。为返回，CephFS 客户端了解文件元数据，可以安全地开始缓存元数据和文件数据。

元数据服务器 (MDS)

MDS 执行以下操作：

- 为 CephFS 客户端提供元数据。
- 管理与 Ceph 文件系统中存储的文件相关的元数据。
- 协调对共享 Red Hat Ceph Storage 的访问。
- 缓存热元数据，以减少对后备元数据池存储的请求。
- 管理 CephFS 客户端的缓存，以维护缓存一致性。
- 在活动 MDS 之间复制热元数据.
- 将元数据变异到压缩日志，并定期刷新到后备元数据池。
- CephFS 要求至少运行一个元数据服务器守护进程 (**ceph-mds**)。

下图显示了 Ceph 文件系统的组件层。



157_Ceph_1021

底层代表底层核心存储集群组件：

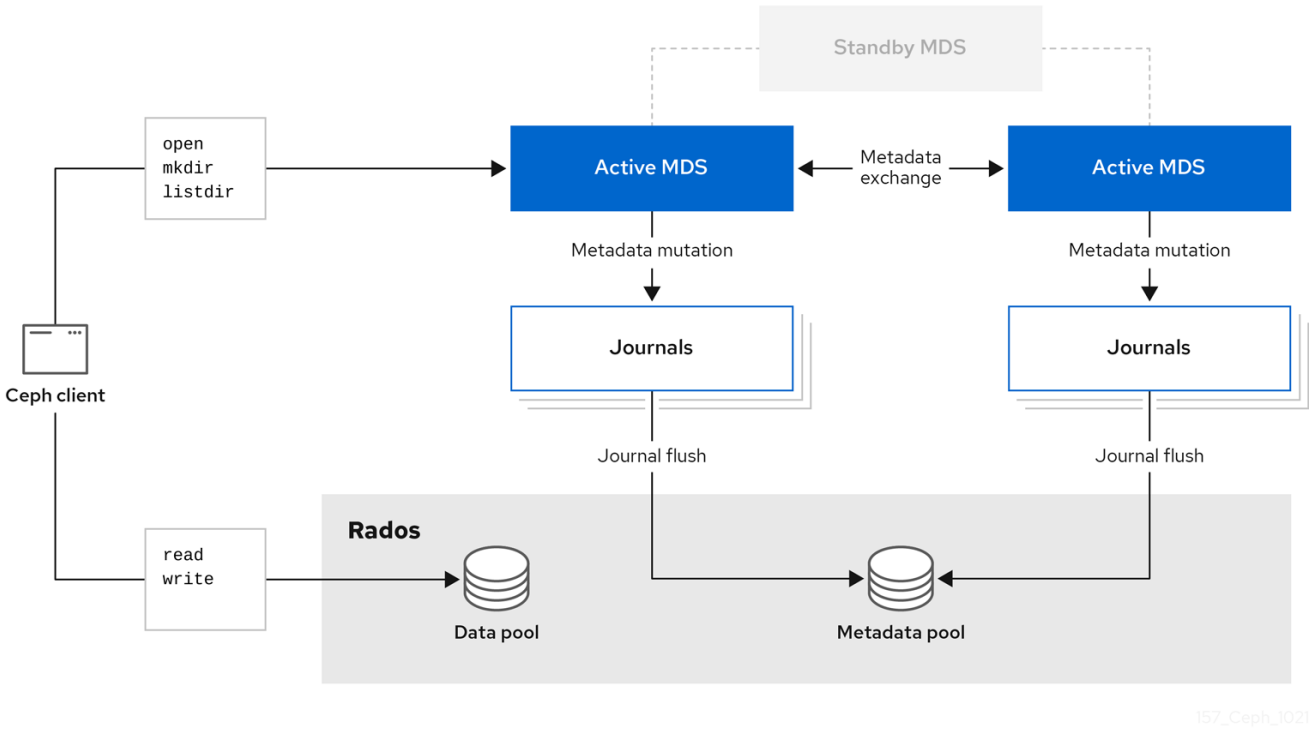
- 存储 Ceph 文件系统数据和元数据的 Ceph OSD (**ceph-osd**)。
- 用于管理 Ceph 文件系统元数据的 Ceph 元数据服务器 (**ceph-mds**)。
- Ceph 监控器 (**ceph-mon**)，用于管理 cluster map 的主副本。

Ceph 存储协议层代表 Ceph 原生 **librados** 库，用于与核心存储集群交互。

CephFS 库层包含 CephFS **libcephfs** 库，它位于 **librados** 基础上，代表 Ceph 文件系统。

顶层代表两种类型的 Ceph 客户端，它们可以访问 Ceph 文件系统。

下图显示了 Ceph 文件系统组件如何相互交互的更多详细信息。



其它资源

- 请参阅*文件系统指南*中的 *Ceph 编排器部分管理 MDS 服务*，以安装 Ceph 元数据服务器。
- 请参阅 *Red Hat Ceph Storage File System Guide* 中的 *Ceph 文件系统部署*一节，以创建 Ceph 文件系统。

1.3. CEPH 文件系统和 SELINUX

从 Red Hat Enterprise Linux 8.3 和 Red Hat Ceph Storage 4.2 开始，支持在 Ceph 文件系统 (CephFS) 环境中使用 Security-Enhanced Linux (SELinux)。现在，您可以使用 CephFS 设置任何 SELinux 文件类型，以及对各个文件分配特定的 SELinux 类型。这一支持适用于 Ceph 文件系统元数据服务器 (MDS)、用户空间 (FUSE) 客户端中的 CephFS 文件系统，以及 CephFS 内核客户端。

其它资源

- 有关 SELinux 的更多信息，请参阅在 Red Hat Enterprise Linux 8 中*使用 SELinux 指南*。

1.4. CEPH 文件系统限制和 POSIX 标准

Ceph 文件系统通过以下方式偏离严格的 POSIX 语义：

- 如果客户端尝试编写文件失败，写入操作不一定是原子的。也就是说，客户端可能会在使用带有 8MB 缓冲区的 **O_SYNC** 标志打开的文件上调用 **write()** 系统调用，然后意外终止写入操作，只能部分应用写入操作。几乎所有文件系统（甚至本地文件系统）都有此行为。
- 在同时发生写操作的情况下，超过对象边界的写入操作不一定是原子的。例如，写入器 *A* 写入 **"aa|aa"** 和 writer *B* 同时写入 **"bb|bb"**，其中 **"|"** 是对象边界，编写 **"aa|bb"** 而不是正确的 **"aa|aa"** 或 **"bb|bb"**。

- POSIX 包含 **telldir()** 和 **searchdir()** 系统调用，允许您获取当前目录偏移并返回该偏移。由于 CephFS 可能会随时对目录进行碎片整理，因此难以为目录返回稳定的整数偏移。因此，调用 **searchdir()** 系统调用到非零偏移通常可以正常工作，但不保证这样做。调用 **seekdir()** 以偏移 0 将始终正常工作。这等同于 **rewinddir()** 系统调用。

- 稀疏文件传播错误地传播到 **stat()** 系统调用的 **st_blocks** 字段。CephFS 不会显式跟踪已分配或写入的文件部分，因为 **st_blocks** 字段始终由按块大小划分的法定文件大小填充。此行为可导致 **du** 等实用程序过量使用的空间。

- 当 **mmap()** 系统调用将文件映射到多个主机上的内存时，写入操作不会被一致的传播到其他主机的缓存中。也就是说，如果页面缓存在主机 A 上，然后在主机 B 上更新，则主机 A 页面不会被统一无效。

- CephFS 客户端提供隐藏的 **.snap** 目录，用于访问、创建、删除和重命名快照。虽然该目录不包括在 **readdir()** 系统调用中，但尝试创建同名文件或目录的任何进程都会返回错误。此隐藏目录的名称可以在挂载时通过 **-o snapdirname=.<new_name>** 选项或使用 **client_snapdir** 配置选项来更改。

其它资源

- 请参阅 *文件系统指南*中的 *Ceph 编排器部分管理 MDS 服务*，以安装 Ceph 元数据服务器。

- 请参阅 *Red Hat Ceph Storage File System Guide* 中的 *Ceph 文件系统部署*一节，以创建 Ceph 文件系统。

其它资源

- 如果要将 NFS Ganesha 用作使用 Red Hat OpenStack Platform 的 Ceph 文件系统的接口，请参阅 *CephFS via NFS Back End Guide for the Shared File System Service* 中的 CephFS with NFS-Ganesha deployment一节，以了解如何通过 NFS 部署此类环境的说明。

第 2 章 CEPH 文件系统元数据服务器

作为存储管理员，您可以了解 Ceph 文件系统 (CephFS) 元数据服务器 (MDS) 的不同状态，以及了解 CephFS MDS 划分机制、配置 MDS 备用守护进程和缓存大小限制。了解这些概念可以让您为存储环境配置 MDS 守护进程。

先决条件

- 一个运行良好、健康的 Red Hat Ceph Storage 集群。
- 安装 Ceph 元数据服务器守护进程 (**ceph-mds**)。有关配置 MDS 守护进程的详细信息，请参阅 *Red Hat Ceph Storage 文件系统指南* 中的*使用 Ceph Orchestrator 来管理 MDS 服务*。

2.1. 元数据服务器守护进程状态

元数据服务器 (MDS) 守护进程在两个状态下运作：

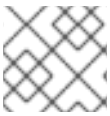
- active SAS-small 管理 Ceph 文件系统中存储的文件和目录的元数据。
- standby SAS- SASserves 作为备份，并在活跃的 MDS 守护进程变得无响应时处于活动状态。

默认情况下，Ceph 文件系统仅使用一个活跃的 MDS 守护进程。但是，具有许多客户端的系统得益于多个活跃的 MDS 守护进程。

您可以将文件系统配置为使用多个活跃 MDS 守护进程，以便您可以扩展更大工作负载的元数据性能。当元数据负载模式发生变化时，活跃的 MDS 守护进程会动态共享元数据工作负载。请注意，具有多个活跃 MDS 守护进程的系统仍需要备用 MDS 守护进程才能保持高可用性。

当活跃 MDS 守护进程出现故障时会发生什么情况

当活跃的 MDS 变得不响应时，Ceph monitor 守护进程会等待与 **mds_beacon_grace** 选项中指定的值相等的秒数。如果在指定的时间段过后活跃的 MDS 仍然不响应，Ceph 监控器会将 MDS 守护进程标记为 **laggy**。其中一个备用守护进程会变为活动状态，具体取决于配置。



注意

若要更改 **mds_beacon_grace** 的值，可添加此选项到 Ceph 配置文件并指定新值。

2.2. 元数据服务器排名

每一 Ceph 文件系统 (CephFS) 具有多个等级，默认为一，从零开始。

等级定义在多个元数据服务器 (MDS) 守护进程之间共享元数据工作负载的方式。等级数是一次可以处于活跃状态的最大 MDS 守护进程数。每个 MDS 守护进程处理分配给该等级的 CephFS 元数据子集。

每个 MDS 守护进程最初启动且没有等级。Ceph 监控器将排名分配到 守护进程。MDS 守护进程一次只能有一个排名。后台程序仅在停止时丢失等级。

max_mds 设置控制将创建等级数。

只有备用守护进程可用于接受新等级时，CephFS 中实际的排名数量才会增加。

等级状态

等级可以是：

- **Up** - 分配给 MDS 守护进程的排名。
- **Failed** - 与任何 MDS 守护进程无关的等级。
- **Damaged** - 损坏的等级；其元数据被损坏或缺失。在操作器修复问题之前，损坏的等级不会分配到任何 MDS 守护进程，并对损坏的等级使用 **ceph mds repaired** 的命令。

2.3. 元数据服务器缓存大小限制

您可以通过以下方法限制 Ceph 文件系统 (CephFS) 元数据服务器 (MDS) 缓存的大小：

- **内存限制** ：使用 **mds_cache_memory_limit** 选项。红帽建议为 **mds_cache_memory_limit** 设置 8 GB 到 64 GB 的值。设置更多缓存可能会导致恢复问题。这个限制是使用 MDS 所需最大内存用量的大约 66%。

注意

mds_cache_memory_limit 的默认值为 4 GB。由于默认值超出了推荐的范围，因此红帽建议在上述范围内设置值。

重要

红帽建议使用内存限值而不是内节点计数限制。

- **索引节点计数** ：使用 **mds_cache_size** 选项.默认情况下，禁用按索引节点计数限制 MDS 缓存。

另外，您还可以为 MDS 操作使用 **mds_cache_reservation** 选项来指定缓存保留。缓存保留是内存或索引节点限制的百分比，默认设置为 5%。此参数的目的是让 MDS 为其缓存保留额外内存，以便使用新的元数据操作。因此，MDS 通常应在内存限制下运行，因为它会从客户端重新调用旧状态，从而在其缓存中丢弃未使用的元数据。

在所有情况下，**mds_cache_reservation** 选项替换 **ds_health_cache_threshold** 选项，但 MDS 节点会向 Ceph monitor 发送健康警报，表示缓存太大。默认情况下，**mds_health_cache_threshold** 是最大缓存大小的 150%。

请注意，缓存限制不是硬限制。CephFS 客户端或 MDS 或 MDS 中潜在的错误行为或行为不当可能会导致 MDS 超过其缓存大小。**mds_health_cache_threshold** 选项配置存储集群健康警告消息，以便操作员可以调查 MDS 无法缩小其缓存的原因。

其它资源

- 如需更多信息，请参阅 *Red Hat Ceph Storage 文件系统指南中的 元数据服务器守护进程配置参考部分*。

2.4. 文件系统关联性

您可以配置 Ceph 文件系统(CephFS)，使其首选使用一个 Ceph MDS 而不是另外一个 Ceph MDS。例如，您希望在一个更新更快的硬件上运行 MDS，而不是在一个备用的、较旧较慢的硬件上运行 MDS。您可以通过设置 **mds_join_fs** 选项来指定这个首选项，该选项强制执行这个文件系统关联性。Ceph Monitor 会向 MDS 备用守护进程提供 **mds_join_fs** 与带有失败等级的文件系统名称相等的 mds_join_fs 守护进程。在选择其他待机后台程序前，会选择待机守护进程。如果没有带有 **mds_join_fs** 选项的待机守护进程，则 Ceph 监控将选择常规备用模式作为最后的手段，或者选择其他备用模式作为最后的手段。Ceph 监控器将定期检查 Ceph 文件系统，以查看具有强度关联性的备用功能，以替换具有较低关联性的 Ceph MDS。

其它资源

- 详情请参阅 *Red Hat Ceph Storage 文件系统指南* 中的*配置文件系统关联性*部分。

2.5. 使用 CEPH ORCHESTRATOR 管理 MDS 服务

作为存储管理员，您可以在后端中将 Ceph Orchestrator 与 Cephadm 搭配使用，以部署 MDS 服务。默认情况下，Ceph 文件系统(CephFS)仅使用了一个活跃的 MDS 守护进程。但是，具有许多客户端的系统得益于多个活跃的 MDS 守护进程。

本节涵盖了以下管理任务：

- *使用命令行界面 部署 MDS 服务。*
- *使用服务规格部署 MDS 服务。*
- *使用 Ceph 编排器移除 MDS 服务。*

先决条件

- 一个正在运行的 Red Hat Ceph Storage 集群。
- 所有节点的根级别访问权限。
- 主机添加到集群中。
- 部署所有管理器、监控和 OSD 守护进程。

2.5.1. 使用命令行界面部署 MDS 服务

通过使用 Ceph 编排器，您可以使用命令行界面中的 **placement** 规格部署元数据服务器(MDS)服务。Ceph 文件系统(CephFS)需要一个或多个 MDS。

注意

确保至少有一个池，一个用于 Ceph 文件系统(CephFS)数据，另一个用于 CephFS 元数据。

先决条件

- 一个正在运行的 Red Hat Ceph Storage 集群。
- 主机添加到集群中。
- 部署所有管理器、监控和 OSD 守护进程。

流程

1. 登录到 Cephadm shell：

   示例

   ```
   [root@host01 ~]# cephadm shell
   ```

2. 使用放置规格部署 MDS 守护进程有两种方法：

方法 1

- 使用 **ceph fs volume** 来创建 MDS 守护进程。这将创建 CephFS 卷和与 CephFS 关联的池，也会在主机上启动 MDS 服务。

  语法

  ```
  ceph fs volume create FILESYSTEM_NAME --placement="NUMBER_OF_DAEMONS HOST_NAME_1 HOST_NAME_2 HOST_NAME_3"
  ```

  注意

  默认情况下，为此命令创建池。

  示例

  ```
  [ceph: root@host01 /]# ceph fs volume create test --placement="2 host01 host02"
  ```

方法 2

- 创建池 CephFS，然后使用放置规格部署 MDS 服务：

  a. 为 CephFS 创建池：

  语法

  ```
  ceph osd pool create DATA_POOL [PG_NUM]
  ceph osd pool create METADATA_POOL [PG_NUM]
  ```

  示例

  ```
  [ceph: root@host01 /]# ceph osd pool create cephfs_data 64
  [ceph: root@host01 /]# ceph osd pool create cephfs_metadata 64
  ```

  通常，元数据池可以从保守的 PG 数量开始，因为它的对象通常比数据池少得多。如果需要，可以增加 PG 数量。池大小范围从 64 个 PG 到 512 个 PG。数据池的大小与您文件系统中预期的文件的编号和大小成比例。

  重要

  对于元数据池，请考虑使用：

  - 更高的复制级别，因为对此池的任何数据丢失都可能会导致整个文件系统无法访问。
  - 延迟较低的存储（如 Solid-State Drive(SSD)磁盘），因为这会直接影响客户端上观察到的文件系统操作延迟。

  b. 为数据池和元数据池创建文件系统：

语法

```
ceph fs new FILESYSTEM_NAME METADATA_POOL DATA_POOL
```

示例

```
[ceph: root@host01 /]# ceph fs new test cephfs_metadata cephfs_data
```

c. 使用 **ceph orch apply** 命令部署 MDS 服务：

语法

```
ceph orch apply mds FILESYSTEM_NAME --placement="NUMBER_OF_DAEMONS HOST_NAME_1 HOST_NAME_2 HOST_NAME_3"
```

示例

```
[ceph: root@host01 /]# ceph orch apply mds test --placement="2 host01 host02"
```

验证

- 列出服务：

  示例

  ```
  [ceph: root@host01 /]# ceph orch ls
  ```

- 检查 CephFS 状态：

  示例

  ```
  [ceph: root@host01 /]# ceph fs ls
  [ceph: root@host01 /]# ceph fs status
  ```

- 列出主机、守护进程和进程：

  语法

  ```
  ceph orch ps --daemon_type=DAEMON_NAME
  ```

  示例

  ```
  [ceph: root@host01 /]# ceph orch ps --daemon_type=mds
  ```

其它资源

- 有关创建 Ceph 文件系统(CephFS)的更多信息，请参阅 *Red Hat Ceph Storage File System Guide*。
- 有关设置池值的详情，请参考 在池中设置放置组数量。

2.5.2. 使用服务规格部署 MDS 服务

通过使用 Ceph 编排器，您可以使用服务规格部署 MDS 服务。

注意

确保至少有两个池，一个用于 Ceph 文件系统(CephFS)数据，另一个用于 CephFS 元数据。

先决条件

- 一个正在运行的 Red Hat Ceph Storage 集群。
- 主机添加到集群中。
- 部署所有管理器、监控和 OSD 守护进程。

流程

1. 创建 **mds.yaml** 文件：

   示例

   ```
   [root@host01 ~]# touch mds.yaml
   ```

2. 编辑 **mds.yaml** 文件，使其包含以下详情：

   语法

   ```
   service_type: mds
   service_id: FILESYSTEM_NAME
   placement:
     hosts:
     - HOST_NAME_1
     - HOST_NAME_2
     - HOST_NAME_3
   ```

   示例

   ```
   service_type: mds
   service_id: fs_name
   placement:
     hosts:
     - host01
     - host02
   ```

3. 将 YAML 文件挂载到容器中的一个目录下：

   示例

   ```
   [root@host01 ~]# cephadm shell --mount mds.yaml:/var/lib/ceph/mds/mds.yaml
   ```

4. 进入该目录：

示例

```
[ceph: root@host01 /]# cd /var/lib/ceph/mds/
```

5. 登录到 Cephadm shell：

示例

```
[root@host01 ~]# cephadm shell
```

6. 进入以下目录：

示例

```
[ceph: root@host01 /]# cd /var/lib/ceph/mds/
```

7. 使用服务规格部署 MDS 服务：

语法

```
ceph orch apply -i FILE_NAME.yaml
```

示例

```
[ceph: root@host01 mds]# ceph orch apply -i mds.yaml
```

8. 部署 MDS 服务后，创建 CephFS：

语法

```
ceph fs new CEPHFS_NAME METADATA_POOL DATA_POOL
```

示例

```
[ceph: root@host01 /]# ceph fs new test metadata_pool data_pool
```

验证

- 列出服务：

示例

```
[ceph: root@host01 /]# ceph orch ls
```

- 列出主机、守护进程和进程：

语法

```
ceph orch ps --daemon_type=DAEMON_NAME
```

示例

```
[ceph: root@host01 /]# ceph orch ps --daemon_type=mds
```

其它资源

- 有关创建 Ceph 文件系统(CephFS)的更多信息，请参阅 *Red Hat Ceph Storage File System Guide*。

2.5.3. 使用 Ceph Orchestrator 删除 MDS 服务

您可以使用 **ceph orch rm** 命令删除该服务。或者，您也可以删除文件系统和相关池。

先决条件

- 一个正在运行的 Red Hat Ceph Storage 集群。
- 所有节点的根级别访问权限。
- 主机添加到集群中。
- 在主机上至少部署一个 MDS 守护进程。

流程

- 可以通过两种方式从集群中移除 MDS 守护进程：

方法 1

- 移除 CephFS 卷、关联的池和服务：
  a. 登录到 Cephadm shell：

  示例

  ```
  [root@host01 ~]# cephadm shell
  ```

  b. 将配置参数 **mon_allow_pool_delete** 设置为 **true** ：

  示例

  ```
  [ceph: root@host01 /]# ceph config set mon mon_allow_pool_delete true
  ```

  c. 删除文件系统：

  语法

  ```
  ceph fs volume rm FILESYSTEM_NAME --yes-i-really-mean-it
  ```

  示例

  ```
  [ceph: root@host01 /]# ceph fs volume rm cephfs-new --yes-i-really-mean-it
  ```

此命令将删除文件系统、其数据和元数据池。它还会尝试使用启用了 **ceph-mgr** Orchestrator 模块来删除 MDS。

方法 2

- 使用 **ceph orch rm** 命令从整个集群中删除 MDS 服务：

  a. 列出服务：

  示例

  ```
  [ceph: root@host01 /]# ceph orch ls
  ```

  b. 删除服务

  语法

  ```
  ceph orch rm SERVICE_NAME
  ```

  示例

  ```
  [ceph: root@host01 /]# ceph orch rm mds.test
  ```

验证

- 列出主机、守护进程和进程：

  语法

  ```
  ceph orch ps
  ```

  示例

  ```
  [ceph: root@host01 /]# ceph orch ps
  ```

其它资源

- 如需更多信息，请参阅 *Red Hat Ceph Storage Operations 指南* 中的*使用命令行界面部分部署 MDS 服务*。
- 如需更多信息，请参阅 *Red Hat Ceph Storage Operations Guide* 中的*使用服务规格部署 MDS 服务*部分。

2.6. 配置文件系统关联性

为特定 Ceph 元数据服务器(MDS)设置 Ceph 文件系统(CephFS)关联性。

先决条件

- 一个健康且运行的 Ceph 文件系统。
- Ceph 监控节点的根级别访问权限.

流程

1. 检查 Ceph 文件系统的当前状态：

   示例

   ```
   [root@mon ~]# ceph fs dump
   dumped fsmap epoch 399
   ...
   Filesystem 'cephfs01' (27)
   ...
   e399
   max_mds 1
   in      0
   up      {0=20384}
   failed
   damaged
   stopped
   ...
   [mds.a{0:20384} state up:active seq 239 addr
   [v2:127.0.0.1:6854/966242805,v1:127.0.0.1:6855/966242805]]

   Standby daemons:

   [mds.b{-1:10420} state up:standby seq 2 addr
   [v2:127.0.0.1:6856/2745199145,v1:127.0.0.1:6857/2745199145]]
   ```

2. 设置文件系统关联性：

   语法

   ```
   ceph config set STANDBY_DAEMON mds_join_fs FILE_SYSTEM_NAME
   ```

   示例

   ```
   [root@mon ~]# ceph config set mds.b mds_join_fs cephfs01
   ```

   在 Ceph MDS 故障转移事件后，文件系统会优先选择设置关联性的待机守护进程。

   示例

   ```
   [root@mon ~]# ceph fs dump
   dumped fsmap epoch 405
   e405
   ...
   Filesystem 'cephfs01' (27)
   ...
   max_mds 1
   in      0
   up      {0=10420}
   failed
   damaged
   stopped
   ...
   ```

```
[mds.b{0:10420} state up:active seq 274 join_fscid=27 addr
[v2:127.0.0.1:6856/2745199145,v1:127.0.0.1:6857/2745199145]] 1

Standby daemons:

[mds.a{-1:10720} state up:standby seq 2 addr
[v2:127.0.0.1:6854/1340357658,v1:127.0.0.1:6855/1340357658]]
```

1 **mds.b** 守护进程现在在文件系统转储输出中带有 **join_fscid=27**。

重要

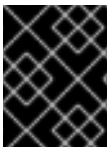
如果文件系统处于降级或非理想状态，则不会执行文件系统关联性。

其它资源

- 请参阅 *Red Hat Ceph Storage 文件系统指南* 中的 *文件系统关联性*部分。

2.7. 配置多个活跃的元数据服务器守护进程

配置多个活动元数据服务器 (MDS) 守护进程，以缩放大型系统的元数据性能。



重要

不要将所有备用 MDS 守护进程转换为活跃的 MDS 守护进程。Ceph 文件系统 (CephFS) 至少需要一个备用 MDS 守护进程才能保持高可用性。

先决条件

- MDS 节点上的 Ceph 管理功能.
- Ceph 监控节点的根级别访问权限.

流程

1. 将 **max_mds** 参数设置为所需的活跃 MDS 守护进程数：

   语法

   ```
   ceph fs set NAME max_mds NUMBER
   ```

   示例

   ```
   [root@mon ~]# ceph fs set cephfs max_mds 2
   ```

   本例将名为 **cephfs**的 CephFS 中的活跃 MDS 守护进程数量增加到两个。

注意

仅当有备用 MDS 守护进程可以取用新等级时，Ceph 才会增加 CephFS 中的实际排名数量。

2. 验证活跃 MDS 守护进程的数量：

语法

```
ceph fs status NAME
```

示例

```
[root@mon ~]# ceph fs status cephfs
cephfs - 0 clients
======
+------+--------+-------+---------------+-------+-------+--------+--------+
| RANK | STATE  |  MDS  |    ACTIVITY   |  DNS  | INOS  |  DIRS  |  CAPS  |
+------+--------+-------+---------------+-------+-------+--------+--------+
|  0   | active | node1 | Reqs:    0 /s |   10  |   12  |   12   |   0    |
|  1   | active | node2 | Reqs:    0 /s |   10  |   12  |   12   |   0    |
+------+--------+-------+---------------+-------+-------+--------+--------+
+-----------------+----------+-------+-------+
|       POOL      |   TYPE   | USED | AVAIL |
+-----------------+----------+-------+-------+
| cephfs_metadata | metadata | 4638  | 26.7G |
|   cephfs_data   |   data   |    0  | 26.7G |
+-----------------+----------+-------+-------+

+-------------+
| STANDBY MDS |
+-------------+
|    node3    |
+-------------+
```

其它资源

- 如需了解更多详细信息，请参阅 *Red Hat Ceph Storage File System Guide* 中的 *Metadata Server daemons states* 章节。
- 如需了解更多详细信息，请参阅 *Red Hat Ceph Storage File System Guide* 中的 *Decreasing the Number of Active MDS Daemons* 部分。
- 如需了解更多详细信息，请参阅 *Red Hat Ceph Storage Administration Guide* 中的 *Managing Ceph users* 章节。

2.8. 配置待机守护进程数量

每个 Ceph 文件系统 (CephFS) 可以指定被视为健康状态所需的待机守护进程数量。这个数字还包括等待排序失败的待机重播守护进程。

先决条件

- Ceph 监控节点的根级别访问权限.

流程

- 为特定 CephFS 设置预期的待机守护进程数量：

语法

```
ceph fs set FS_NAME standby_count_wanted NUMBER
```

注意

将 *NUMBER* 设置为零可禁用守护进程健康检查。

示例

```
[root@mon ~]# ceph fs set cephfs standby_count_wanted 2
```

这个示例将预期的备用守护进程数设置为 2。

2.9. 配置待机重播元数据服务器

通过添加待机重播元数据服务器 (MDS) 守护进程来配置各个 Ceph 文件系统 (CephFS)。如果活跃 MDS 不可用，可以缩短故障切换时间。

这个特定的待机重播守护进程跟踪活跃 MDS 的元数据日志。待机重播守护进程仅供同一排名相同的活跃 MDS 使用，不可用于其他等级。

重要

如果使用 standby-replay，则每个活跃 MDS 都必须具有待机守护进程。

先决条件

- Ceph 监控节点的根级别访问权限.

流程

- 设置特定 CephFS 的待机重播：

  语法

  ```
  ceph fs set FS_NAME allow_standby_replay 1
  ```

  示例

  ```
  [root@mon ~]# ceph fs set cephfs allow_standby_replay 1
  ```

  在本例中，布尔值为 **1**，它允许将 standby-replay 守护进程分配到活动的 Ceph MDS 守护进程。

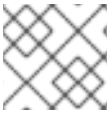
其它资源

- 详情请参阅 *Red Hat Ceph Storage 文件系统指南中的 使用 ceph mds fail 命令* 一节。

2.10. 临时固定策略

临时固定是一个子树的静态分区，可以使用扩展属性来设置策略。策略可以自动将临时固定到目录。当将

临时固定设置为某个目录时，它会自动分配到一个特定等级，因为在所有 Ceph MDS 等级间统一分布。确定分配哪些等级由一致的哈希和目录的索引节点编号完成。当目录的内节点从文件系统缓存中丢弃时，临时固定不会保留。当 Ceph 元数据服务器(MDS)出现故障时，临时固定将记录在日志中，因此 Ceph MDS 备用服务器不会丢失此信息。使用临时固定的策略有两种：



注意

必须为临时固定策略安装 **attr** 和 **jq** 软件包。

分布式

此策略强制所有目录的即时子对象都必须临时固定。例如，使用分布式策略将用户的主目录分散到整个 Ceph 文件系统集群中。通过设置 **ceph.dir.pin.distributed** 扩展属性来启用此策略。

语法

```
setfattr -n ceph.dir.pin.distributed -v 1 DIRECTORY_PATH
```

示例

```
[root@host01 mount]# setfattr -n ceph.dir.pin.distributed -v 1 dir1/
```

随机

此策略强制实施一个机会，任何子子目录都可能会被临时固定。您可以自定义可临时固定的目录百分比。通过设置 **ceph.dir.pin.random** 并设置百分比来启用此策略。红帽建议将此百分比设置为小于 1% 的值(**0.01**)。太多子树分区可能会导致性能下降。您可以通过设置 **mds_export_ephemeral_random_max** Ceph MDS 配置选项来设置最大百分比。已启用参数 **mds_export_ephemeral_distributed** 和 **mds_export_ephemeral_random**。

语法

```
setfattr -n ceph.dir.pin.random -v PERCENTAGE_IN_DECIMAL DIRECTORY_PATH
```

示例

```
[root@host01 mount]# setfattr -n ceph.dir.pin.random -v  0.01 dir1/
```

启用固定后，您可以通过运行以下命令来 验证 ：

语法

```
getfattr -n ceph.dir.pin.random DIRECTORY_PATH
getfattr -n ceph.dir.pin.distributed DIRECTORY_PATH
```

示例

```
[root@host01 mount]# getfattr -n ceph.dir.pin.distributed dir1/
# file: dir1/
ceph.dir.pin.distributed="1"
```

```
[root@host01 mount]# getfattr -n ceph.dir.pin.random dir1/
# file: dir1/
ceph.dir.pin.random="0.01"
```

示例

```
[ceph: root@host01 /]# ceph tell mds.a get subtrees | jq '.[] | [.dir.path, .auth_first, .export_pin]'
```

如果目录已被固定，则 **export_pin 的值为** 0 （如果它固定到排名 **0**），**1** （如果它固定到排名 1），以此类推。如果目录没有固定，则值为 **-1**。

要删除 分区策略，请删除扩展属性或将值设为 **0。**

语法

```
setfattr -n ceph.dir.pin.distributed -v 0 DIRECTORY_PATH
```

示例

```
[root@host01 mount]# setfattr -n ceph.dir.pin.distributed -v 0 dir1/
```

您可以通过 **运行以下命令来**验证。Syntax

```
getfattr -n ceph.dir.pin.distributed DIRECTORY_PATH
```

示例

```
[root@host01 mount]# getfattr -n ceph.dir.pin.distributed dir1/
```

对于导出固定，请删除扩展属性，或者将扩展属性设置为 **-1**。

语法

```
setfattr -n ceph.dir.pin -v -1 DIRECTORY_PATH
```

示例

```
[root@host01 mount]# setfattr -n ceph.dir.pin -v -1 dir1/
```

其它资源

- 有关手动设置固定的详细信息，请参阅 *Red Hat Ceph Storage 文件系统指南* 中的*手动固定目录树* 部分。

2.11. 手动将目录树固定到特定等级

有时，可能需要使用显式将元数据映射到特定 Ceph 元数据服务器(MDS)等级来覆盖动态均衡器。您可以手动执行此操作，以平均分配应用的负载，或者限制用户对 Ceph 文件系统集群的元数据请求的影响。通过设置 **ceph.dir.pin** 扩展属性，手动固定目录也称为导出固定。

目录的导出固定从最接近的父目录继承，但可通过在该目录中设置导出固定来覆盖。在目录上设置导出固定会影响所有子目录，例如：

```
[root@client ~]# mkdir -p a/b 1
[root@client ~]# setfattr -n ceph.dir.pin -v 1 a/ 2
[root@client ~]# setfattr -n ceph.dir.pin -v 0 a/b 3
```

1. 目录 **a/** 和 **a/b** 都会在没有导出固定设置的情况下启动。
2. 目录 **a/** 和 **a/b** 现在固定到 rank **1**。
3. 现在，目录 **a/b** 被固定到 rank **0**，目录 **a/** 和它的剩余子目录被固定到 rank **1**。

先决条件

- 一个正在运行的 Red Hat Ceph Storage 集群。
- 正在运行的 Ceph 文件系统。
- CephFS 客户端的 root 级别访问权限。
- 安装 **attr** 软件包。

流程

- 在目录中设置导出固定：

  语法

  ```
  setfattr -n ceph.dir.pin -v RANK PATH_TO_DIRECTORY
  ```

  示例

  ```
  [root@client ~]# setfattr -n ceph.dir.pin -v 2 cephfs/home
  ```

其它资源

- 如需了解有关自动设置 pin 的详细信息，请参阅 *Red Hat Ceph Storage File System Guide* 中的*临时固定策略*部分。

2.12. 减少活跃元数据服务器守护进程数量

如何减少活动 Ceph 文件系统 (CephFS) 元数据服务器 (MDS) 守护进程的数量。

先决条件

- 要删除的等级必须首先处于活跃状态，这意味着您必须具有与 **max_mds** 参数指定的 MDS 守护进程数量相同的 MDS 守护进程数。
- Ceph 监控节点的根级别访问权限.

流程

1. 设置由 **max_mds** 参数指定的相同 MDS 守护进程数：

   语法

   ```
   ceph fs status NAME
   ```

   示例

   ```
   [root@mon ~]# ceph fs status cephfs
   cephfs - 0 clients

   +------+--------+-------+---------------+-------+-------+--------+--------+
   | RANK | STATE  |  MDS  |    ACTIVITY   |  DNS  | INOS |  DIRS  |  CAPS  |
   +------+--------+-------+---------------+-------+-------+--------+--------+
   |  0   | active | node1 | Reqs:    0 /s |   10  |   12  |   12   |   0    |
   |  1   | active | node2 | Reqs:    0 /s |   10  |   12  |   12   |   0    |
   +------+--------+-------+---------------+-------+-------+--------+--------+
   +-----------------+----------+-------+-------+
   |       POOL      |   TYPE   |  USED | AVAIL |
   +-----------------+----------+-------+-------+
   | cephfs_metadata | metadata | 4638  | 26.7G |
   |   cephfs_data   |   data   |    0  | 26.7G |
   +-----------------+----------+-------+-------+

   +-------------+
   | Standby MDS |
   +-------------+
   |    node3    |
   +-------------+
   ```

2. 在具有管理功能的节点中，将 **max_mds** 参数改为所需的活跃 MDS 守护进程数：

   语法

   ```
   ceph fs set NAME max_mds NUMBER
   ```

   示例

   ```
   [root@mon ~]# ceph fs set cephfs max_mds 1
   ```

3. 通过观察 Ceph 文件系统状态，等待存储集群稳定到新的 **max_mds** 值。

4. 验证活跃 MDS 守护进程的数量：

   语法

   ```
   ceph fs status NAME
   ```

   示例

   ```
   [root@mon ~]# ceph fs status cephfs
   cephfs - 0 clients
   ```

```
+------+--------+-------+--------------+-------+-------+--------+--------+
| RANK | STATE  |  MDS  |    ACTIVITY  | DNS   | INOS  | DIRS   | CAPS   |
+------+--------+-------+--------------+-------+-------+--------+--------+
|  0   | active | node1 | Reqs:  0 /s  |  10   |  12   |  12    |  0     |
+------+--------+-------+--------------+-------+-------+--------|--------+
+-----------------+----------+-------+-------+
|       POOl      |   TYPE   | USED | AVAIL |
+-----------------+----------+-------+-------+
| cephfs_metadata | metadata | 4638  | 26.7G |
|   cephfs_data   |   data   |   0   | 26.7G |
+-----------------+----------+-------+-------+

+-------------+
| Standby MDS |
+-------------+
|    node3    |
|    node2    |
+-------------+
```

其它资源

- 请参阅 *Red Hat Ceph Storage 文件系统指南中的 元数据服务器守护进程状态* 部分。
- 如需更多信息，请参阅 *Red Hat Ceph Storage 文件系统指南* 中的*配置多个活跃元数据服务器守护进程* 部分。
- 有关安装 *Red Hat Ceph Storage 集群的详细信息，请参阅 Red Hat Ceph Storage 安装指南* 。

2.13. 查看 CEPH 元数据服务器客户端的指标

您可以使用命令行界面查看 Ceph 元数据服务器(MDS)的指标。CephFS 使用 **Perf Counters** 来跟踪指标。您可以使用 计数器 dump 命令查看指标。

先决条件

- 正在运行的 IBM Storage Ceph 集群。

流程

1. 获取 **mds** 服务的名称：

   语法

   ```
   [ceph: root@mds-host01 /]# ceph orch ps | grep mds
   ```

2. 检查每个客户端指标的 MDS：

   语法

   ```
   [ceph: root@mds-host01 /]# ceph tell MDS_SERVICE_NAME counter dump
   ```

   示例

```
[root@ceph2-hk-n-0mfqao-node1-installer ~]# ceph tell mds.cephfs.ceph2-hk-n-0mfqao-node4.isztbk counter dump

[
  {
    "key": "mds_client_metrics",
    "value": [
      {
        "labels": {
          "fs_name": "cephfs",
          "id": "24379"
        },
        "counters": {
          "num_clients": 4
        }
      }
    ]
  },
  {
    "key": "mds_client_metrics-cephfs",
    "value": [
      {
        "labels": {
          "client": "client.24413",
          "rank": "0"
        },
        "counters": {
          "cap_hits": 56,
          "cap_miss": 9,
          "avg_read_latency": 0E-9,
          "avg_write_latency": 0E-9,
          "avg_metadata_latency": 0E-9,
          "dentry_lease_hits": 2,
          "dentry_lease_miss": 12,
          "opened_files": 0,
          "opened_inodes": 9,
          "pinned_icaps": 4,
          "total_inodes": 9,
          "total_read_ops": 0,
          "total_read_size": 0,
          "total_write_ops": 0,
          "total_write_size": 0
        }
      },
      {
        "labels": {
          "client": "client.24502",
          "rank": "0"
        },
        "counters": {
          "cap_hits": 921403,
          "cap_miss": 102382,
          "avg_read_latency": 0E-9,
          "avg_write_latency": 0E-9,
          "avg_metadata_latency": 0E-9,
          "dentry_lease_hits": 17117,
```

```
    "dentry_lease_miss": 204710,
    "opened_files": 0,
    "opened_inodes": 9,
    "pinned_icaps": 7,
    "total_inodes": 9,
    "total_read_ops": 0,
    "total_read_size": 0,
    "total_write_ops": 1,
    "total_write_size": 132
   }
  },
  {
   "labels": {
    "client": "client.24508",
    "rank": "0"
   },
   "counters": {
    "cap_hits": 928694,
    "cap_miss": 103183,
    "avg_read_latency": 0E-9,
    "avg_write_latency": 0E-9,
    "avg_metadata_latency": 0E-9,
    "dentry_lease_hits": 17217,
    "dentry_lease_miss": 206348,
    "opened_files": 0,
    "opened_inodes": 9,
    "pinned_icaps": 7,
    "total_inodes": 9,
    "total_read_ops": 0,
    "total_read_size": 0,
    "total_write_ops": 1,
    "total_write_size": 132
   }
  },
  {
   "labels": {
    "client": "client.24520",
    "rank": "0"
   },
   "counters": {
    "cap_hits": 56,
    "cap_miss": 9,
    "avg_read_latency": 0E-9,
    "avg_write_latency": 0E-9,
    "avg_metadata_latency": 0E-9,
    "dentry_lease_hits": 2,
    "dentry_lease_miss": 12,
    "opened_files": 0,
    "opened_inodes": 9,
    "pinned_icaps": 4,
    "total_inodes": 9,
    "total_read_ops": 0,
    "total_read_size": 0,
    "total_write_ops": 0,
    "total_write_size": 0
   }
```

```
      }
    ]
  }
]
```

客户端指标描述

CephFS 将客户端指标导出为 Labeled Perf Counters，您可以使用它来监控客户端的性能。CephFS 导出以下客户端指标：

NAME	TYPE	DESCRIPTION
cap_hits	量表	文件能力百分比超过 caps 总数。
cap_miss	量表	文件功能的百分比丢失超过 caps 总数。
avg_read_latency	量表	平均读取延迟的值。
avg_write_latency	量表	写入延迟的值。
avg_metadata_latency	量表	元数据延迟的值
dentry_lease_hits	量表	超过总 dentry 租期请求的 dentry 租期命中的百分比。
dentry_lease_miss	量表	dentry 租期丢失的百分比超过总 dentry 租期请求。
open_files	量表	已打开的文件数。
opened_inodes	量表	已打开的索引节点数.
pinned_icaps	量表	固定 Inode Caps 的数量。
total_inodes	量表	节点总数.
total_read_ops	量表	所有进程生成的读取操作总数。
total_read_size	量表	所有进程生成的输入/输出操作中的读取字节数。
total_write_ops	量表	所有进程生成的写入操作总数。
total_write_size	量表	使用所有进程生成的输入/输出操作写入的字节数。

第 3 章 CEPH 文件系统的部署

作为存储管理员，您可以在存储环境中部署 Ceph 文件系统 (CephFS)，并让客户端挂载这些 Ceph 文件系统来满足存储需求。

基本上，部署工作流有三个步骤：

1. 在 Ceph 监控节点上创建 Ceph 文件系统。
2. 创建具有适当功能的 Ceph 客户端用户，并在将要挂载 Ceph 文件系统的节点上提供客户端密钥。
3. 使用内核客户端或用户空间 (FUSE) 客户端中的文件系统，将 CephFS 挂载到专用节点上。

先决条件

- 一个运行良好、健康的 Red Hat Ceph Storage 集群。
- 安装和配置 Ceph 元数据服务器守护进程 (**ceph-mds**)。

3.1. 布局、配额、快照和网络限制

这些用户功能可以帮助您根据所需的要求限制对 Ceph 文件系统 (CephFS) 的访问。

重要

除 **rw** 外，所有用户能力标志都必须按字母顺序指定。

布局和配额

使用布局或配额时，除了 **rw** 功能外，客户端还需要 **p** 标志。设置 **p** 标志会限制由特殊扩展属性（带有 **ceph.** 前缀）设置的所有属性。另外，这限制了设置这些字段的其他方法，如 **openc** 操作使用布局。

示例

```
client.0
    key: AQAz7EVWygILFRAAdIcuJ10opU/JKyfFmxhuaw==
    caps: [mds] allow rwp
    caps: [mon] allow r
    caps: [osd] allow rw tag cephfs data=cephfs_a

client.1
    key: AQAz7EVWygILFRAAdIcuJ11opU/JKyfFmxhuaw==
    caps: [mds] allow rw
    caps: [mon] allow r
    caps: [osd] allow rw tag cephfs data=cephfs_a
```

在本例中，**client.0** 可以修改文件系统 **cephfs_a** 上的布局和配额，但 **client.1** 无法修改。

快照

在创建或删除快照时，除了 **rw** 功能外，客户端还需要 **s** 标志。当能力字符串还包含 **p** 标志时，**s** 标志必须出现在它后面。

示例

```
client.0
    key: AQAz7EVWygILFRAAdIcuJ10opU/JKyfFmxhuaw==
    caps: [mds] allow rw, allow rws path=/temp
    caps: [mon] allow r
    caps: [osd] allow rw tag cephfs data=cephfs_a
```

在本例中，**client.0** 可以在文件系统 **cephfs_a** 的 **temp** 目录中创建或删除快照。

Network

限制从特定网络连接的客户端.

示例

```
client.0
  key: AQAz7EVWygILFRAAdIcuJ10opU/JKyfFmxhuaw==
  caps: [mds] allow r network 10.0.0.0/8, allow rw path=/bar network 10.0.0.0/8
  caps: [mon] allow r network 10.0.0.0/8
  caps: [osd] allow rw tag cephfs data=cephfs_a network 10.0.0.0/8
```

可选的网络和前缀长度为 CIDR 表示法，例如 **10.3.0.0/16**。

其它资源

- 如需了解有关设置 Ceph 用户功能的详细信息，请参阅 *Red Hat Ceph Storage File System Guide* 中的*Creating client users for a Ceph File System* 部分。

3.2. 创建 CEPH 文件系统

您可以在 Ceph 监控节点上创建多个 Ceph 文件系统(CephFS)。

先决条件

- 一个运行良好、健康的 Red Hat Ceph Storage 集群。
- 安装和配置 Ceph 元数据服务器守护进程 (**ceph-mds**)。
- Ceph 监控节点的根级别访问权限.
- Ceph 客户端节点的根级别访问权限。

流程

1. 配置客户端节点，以使用 Ceph 存储群集。
   a. 启用 Red Hat Ceph Storage Tools 存储库：

   Red Hat Enterprise Linux 8

   ```
   [root@client01 ~]# subscription-manager repos --enable=rhceph-6-tools-for-rhel-8-x86_64-rpms
   ```

   Red Hat Enterprise Linux 9

```
[root@client01 ~]# subscription-manager repos --enable=rhceph-6-tools-for-rhel-9-x86_64-rpms
```

b. 安装 **ceph-fuse** 软件包：

```
[root@client ~]# dnf install ceph-fuse
```

c. 将 Ceph 客户端密钥环从 Ceph 监控节点复制到客户端节点：

语法

```
scp root@MONITOR_NODE_NAME:/etc/ceph/KEYRING_FILE /etc/ceph/
```

将 *MONITOR_NODE_NAME* 替换为 Ceph Monitor 主机名或 IP 地址。

示例

```
[root@client ~]# scp root@192.168.0.1:/etc/ceph/ceph.client.1.keyring /etc/ceph/
```

d. 将 Ceph 配置文件从 Ceph 监控节点复制到客户端节点：

语法

```
scp root@MONITOR_NODE_NAME:/etc/ceph/ceph.conf /etc/ceph/ceph.conf
```

将 *MONITOR_NODE_NAME* 替换为 Ceph Monitor 主机名或 IP 地址。

示例

```
[root@client ~]# scp root@192.168.0.1:/etc/ceph/ceph.conf /etc/ceph/ceph.conf
```

e. 为配置文件设置适当的权限：

```
[root@client ~]# chmod 644 /etc/ceph/ceph.conf
```

2. 创建 Ceph 文件系统：

语法

```
ceph fs volume create FILE_SYSTEM_NAME
```

示例

```
[root@mon ~]# ceph fs volume create cephfs01
```

重复此步骤以创建额外的文件系统。

注意

通过运行此命令，Ceph 会自动创建新的池，并部署新的 Ceph 元数据服务器 (MDS)守护进程来支持新文件系统。这也相应地配置 MDS 关联性。

3. 从 Ceph 客户端验证对新 Ceph 文件系统的访问。

   a. 授权 Ceph 客户端访问新文件系统：

      语法

      ```
      ceph fs authorize FILE_SYSTEM_NAME CLIENT_NAME DIRECTORY PERMISSIONS
      ```

      示例

      ```
      [root@mon ~]# ceph fs authorize cephfs01 client.1 / rw
      [client.1]
          key = BQAmthpf81M+JhAAiHDYQkMiCq3x+J0n9e8REK==

      [root@mon ~]# ceph auth get client.1
      exported keyring for client.1
      [client.1]
          key = BQAmthpf81M+JhAAiHDYQkMiCq3x+J0n9e8REK==
          caps mds = "allow rw fsname=cephfs01"
          caps mon = "allow r fsname=cephfs01"
          caps osd = "allow rw tag cephfs data=cephfs01"
      ```

      注意

      另外，您可以通过指定 **root_squash** 选项来添加安全措施。这可防止禁止带有 **uid=0** 或 **gid=0** 的客户端进行写入操作，但仍然允许读操作意外删除场景。

      示例

      ```
      [root@mon ~]# ceph fs authorize cephfs01 client.1 / rw root_squash
      /volumes rw
      [client.1]
          key = BQAmthpf81M+JhAAiHDYQkMiCq3x+J0n9e8REK==

      [root@mon ~]# ceph auth get client.1
      [client.1]
          key = BQAmthpf81M+JhAAiHDYQkMiCq3x+J0n9e8REK==
          caps mds = "allow rw fsname=cephfs01 root_squash, allow rw
      fsname=cephfs01 path=/volumes"
          caps mon = "allow r fsname=cephfs01"
          caps osd = "allow rw tag cephfs data=cephfs01"
      ```

      在本例中，除非在 **/volumes** 目录树中，为文件系统 **cephfs01** 启用了 **root_squash**。

      重要

      Ceph 客户端只能看到它已被授权的 CephFS。

   b. 将 Ceph 用户的密钥环复制到 Ceph 客户端节点上：

      语法

```
ceph auth get CLIENT_NAME > OUTPUT_FILE_NAME
scp OUTPUT_FILE_NAME TARGET_NODE_NAME:/etc/ceph
```

示例

```
[root@mon ~]# ceph auth get client.1 > ceph.client.1.keyring
exported keyring for client.1
[root@mon ~]# scp ceph.client.1.keyring client:/etc/ceph
root@client's password:
ceph.client.1.keyring                    100%  178   333.0KB/s   00:00
```

c. 在 Ceph 客户端节点上，创建一个新目录：

语法

```
mkdir PATH_TO_NEW_DIRECTORY_NAME
```

示例

```
[root@client ~]# mkdir /mnt/mycephfs
```

d. 在 Ceph 客户端节点上，挂载新的 Ceph 文件系统：

语法

```
ceph-fuse PATH_TO_NEW_DIRECTORY_NAME -n CEPH_USER_NAME --client-fs=_FILE_SYSTEM_NAME
```

示例

```
[root@client ~]# ceph-fuse /mnt/mycephfs/ -n client.1 --client-fs=cephfs01
ceph-fuse[555001]: starting ceph client
2022-05-09T07:33:27.158+0000 7f11feb81200 -1 init, newargv = 0x55fc4269d5d0
newargc=15
ceph-fuse[555001]: starting fuse
```

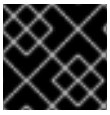
e. 在 Ceph 客户端节点上，列出新挂载点的目录的内容，或者在新挂载点上创建文件。

其它资源

- 如需更多详细信息，请参阅 *Red Hat Ceph Storage 文件系统指南* 中的*为 Ceph 文件系统创建客户端*部分。
- 如需了解更多详细信息，请参阅 *Red Hat Ceph Storage 文件系统指南* 中的*将 Ceph 文件系统挂载为内核客户端*部分。
- 如需了解更多详细信息，请参阅 *Red Hat Ceph Storage 文件系统指南* 中的*将 Ceph 文件系统挂载为 FUSE 客户端*部分。
- 如需了解更多详细信息，请参阅 *Red Hat Ceph Storage 文件系统指南* 中的 *Ceph 文件系统限制和 POSIX 标准* 部分。
- 如需了解更多详细信息，请参见 *Red Hat Ceph Storage 策略指南* 中的 *池* 一章。

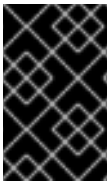
3.3. 将纠删代码池添加到 CEPH 文件系统

默认情况下，Ceph 将复制池用于数据池。若有需要，您还可以向 Ceph 文件系统添加额外的擦除数据池。与由复制池支持的 Ceph 文件系统相比，由纠删代码池支持的 Ceph 文件系统 (CephFS) 使用较少的总存储。尽管纠删代码池使用较少的总存储，它们也使用的内存和处理器资源要多于复制池。



重要

CephFS EC 池仅用于归档目的。



重要

对于生产环境，红帽建议为 CephFS 使用默认复制数据池。在 CephFS 中创建内节点，在默认数据池中至少创建一个对象。最好将复制池用于默认数据，以提高小对象写入性能，并提高更新后端的读取性能。

先决条件

- 一个正在运行的 Red Hat Ceph Storage 集群。
- 现有的 Ceph 文件系统.
- 使用 BlueStore OSD 的池。
- Ceph 监控节点的根级别访问权限.
- 安装 **attr** 软件包。

流程

1. 为 CephFS 创建纠删代码数据池：

   语法

   ```
   ceph osd pool create DATA_POOL_NAME erasure
   ```

   示例

   ```
   [root@mon ~]# ceph osd pool create cephfs-data-ec01 erasure
   pool 'cephfs-data-ec01' created
   ```

2. 验证是否已添加池：

   示例

   ```
   [root@mon ~]# ceph osd lspools
   ```

3. 在纠删代码池中启用覆盖：

   语法

   ```
   ceph osd pool set DATA_POOL_NAME allow_ec_overwrites true
   ```

示例

```
[root@mon ~]# ceph osd pool set cephfs-data-ec01 allow_ec_overwrites true
set pool 15 allow_ec_overwrites to true
```

4. 验证 Ceph 文件系统的状态：

语法

```
ceph fs status FILE_SYSTEM_NAME
```

示例

```
[root@mon ~]# ceph fs status cephfs-ec
cephfs-ec - 14 clients
=========
RANK  STATE            MDS              ACTIVITY     DNS    INOS   DIRS   CAPS
 0    active  cephfs-ec.example.ooymyq  Reqs:    0 /s  8231   8233    891    921
       POOL          TYPE     USED  AVAIL
cephfs-metadata-ec  metadata   787M  8274G
 cephfs-data-ec      data    2360G  12.1T

      STANDBY MDS
cephfs-ec.example.irsrql
cephfs-ec.example.cauuaj
```

5. 将擦除代码的数据池添加到现有的 CephFS 中：

语法

```
ceph fs add_data_pool FILE_SYSTEM_NAME DATA_POOL_NAME
```

示例

```
[root@mon ~]# ceph fs add_data_pool cephfs-ec cephfs-data-ec01
```

本例将新数据池 **cephfs-data-ec01** 添加到现有的 erasure-code 文件系统 **cephfs-ec** 中。

6. 验证纠删代码池是否已添加到 Ceph 文件系统中：

语法

```
ceph fs status FILE_SYSTEM_NAME
```

示例

```
[root@mon ~]# ceph fs status cephfs-ec
cephfs-ec - 14 clients
=========
RANK  STATE            MDS              ACTIVITY     DNS    INOS   DIRS   CAPS
 0    active  cephfs-ec.example.ooymyq  Reqs:    0 /s  8231   8233    891    921
       POOL          TYPE     USED  AVAIL
```

```
cephfs-metadata-ec  metadata   787M  8274G
 cephfs-data-ec      data    2360G  12.1T
cephfs-data-ec01     data       0   12.1T

      STANDBY MDS
cephfs-ec.example.irsrql
cephfs-ec.example.cauuaj
```

7. 在新目录中设置文件布局：

语法

```
mkdir PATH_TO_DIRECTORY
setfattr -n ceph.dir.layout.pool -v DATA_POOL_NAME PATH_TO_DIRECTORY
```

示例

```
[root@mon ~]# mkdir /mnt/cephfs/newdir
[root@mon ~]# setfattr -n ceph.dir.layout.pool -v cephfs-data-ec01 /mnt/cephfs/newdir
```

在本例中，在 **/mnt/cephfs/newdir** 目录中创建的所有新文件都会继承目录布局，并将数据放在新添加的纠删代码池中。

其它资源

- 有关 CephFS MDS 的更多信息，请参阅 *Red Hat Ceph Storage File System Guide* 中的 *The Ceph File System Metadata Server* 章节。
- 如需更多信息，请参阅 *Red Hat Ceph Storage 文件系统指南* 中的*创建 Ceph 文件系统*部分。
- 如需更多信息，请参阅 *Red Hat Ceph Storage 策略指南中的 Erasure Code* 池 一章。
- 如需更多信息，请参阅 *Red Hat Ceph Storage 策略指南中的 使用覆盖* 部分。

3.4. 为 CEPH 文件系统创建客户端用户

Red Hat Ceph Storage 使用 **cephx** 进行身份验证，这在默认情况下是启用的。若要将 **cephx** 与 Ceph 文件系统搭配使用，请在 Ceph 监控节点上创建具有正确授权功能的用户，并在将要挂载 Ceph 文件系统的节点上提供其密钥。

先决条件

- 一个正在运行的 Red Hat Ceph Storage 集群。
- 安装和配置 Ceph 元数据服务器守护进程 (ceph-mds)。
- Ceph 监控节点的根级别访问权限.
- Ceph 客户端节点的根级别访问权限。

流程

1. 在监控节点上登录到 Cephadm shell：

示例

```
[root@host01 ~]# cephadm shell
```

2. 在 Ceph 监控节点上，创建一个客户端用户：

语法

```
ceph fs authorize FILE_SYSTEM_NAME client.CLIENT_NAME /DIRECTORY CAPABILITY
[/DIRECTORY CAPABILITY] PERMISSIONS ...
```

- 将客户端限制为仅在文件系统 **cephfs_a** 的 **temp** 目录中写入：

示例

```
[ceph: root@host01 /]# ceph fs authorize cephfs_a client.1 / r /temp rw

client.1
  key = AQBSdFhcGZFUDRAAcKhG9Cl2HPiDMMRv4DC43A==
```

- 要将客户端完全限制到 **temp** 目录，请删除根 (/) 目录：

示例

```
[ceph: root@host01 /]# ceph fs authorize cephfs_a client.1 /temp rw
```

注意

提供 **all** 或星号作为文件系统名称将授予对每个文件系统的访问权限。通常，需要对星号加上引号以避免它在 shell 中被错误使用。

3. 验证创建的密钥：

语法

```
ceph auth get client.ID
```

示例

```
[ceph: root@host01 /]# ceph auth get client.1

client.1
  key = AQBSdFhcGZFUDRAAcKhG9Cl2HPiDMMRv4DC43A==
  caps mds = "allow r, allow rw path=/temp"
  caps mon = "allow r"
  caps osd = "allow rw tag cephfs data=cephfs_a"
```

4. 将密钥环复制到客户端。

a. 在 Ceph 监控节点上，将密钥环导出到文件中：

语法

```
ceph auth get client.ID -o ceph.client.ID.keyring
```

示例

```
[ceph: root@host01 /]# ceph auth get client.1 -o ceph.client.1.keyring
exported keyring for client.1
```

b. 将 Ceph 监控节点的客户端密钥环复制到客户端节点上的 **/etc/ceph/** 目录中：

语法

```
scp /ceph.client.ID.keyring root@CLIENT_NODE_NAME:/etc/ceph/ceph.client.ID.keyring
```

将 *CLIENT_NODE_NAME* 替换为 Ceph 客户端节点名称或 IP。

示例

```
[ceph: root@host01 /]# scp /ceph.client.1.keyring
root@client01:/etc/ceph/ceph.client.1.keyring
```

5. 在客户端节点中，为密钥环文件设置适当的权限：

语法

```
chmod 644 ceph.client.ID.keyring
```

示例

```
[root@client01 ~]# chmod 644 /etc/ceph/ceph.client.1.keyring
```

其它资源

- 如需了解更多详细信息，请参见 *Red Hat Ceph Storage 管理指南* 中的 *用户管理*一章。

3.5. 将 CEPH 文件系统挂载为内核客户端

您可以将 Ceph 文件系统 (CephFS) 挂载为内核客户端，也可以手动挂载或在系统引导时自动挂载。

重要

除了 Red Hat Enterprise Linux 外，还允许在其他 Linux 发行版上运行的客户端，但不受支持。如果在 CephFS 元数据服务器或存储群集的其他部分使用这些客户端，红帽会解决这些问题。如果发现原因在客户端，则该问题必须由 Linux 发行版的内核供应商解决。

先决条件

- 对基于 Linux 的客户端节点的根级别访问权限.
- Ceph 监控节点的根级别访问权限.
- 现有的 Ceph 文件系统.

流程

1. 配置客户端节点，以使用 Ceph 存储群集。

   a. 启用 Red Hat Ceph Storage 7 Tools 存储库：

      Red Hat Enterprise Linux 9

      ```
      [root@client01 ~]# subscription-manager repos --enable=rhceph-6-tools-for-rhel-9-x86_64-rpms
      ```

   b. 安装 **ceph-common** 软件包：

      ```
      [root@client01 ~]# dnf install ceph-common
      ```

   c. 在监控节点上登录到 Cephadm shell：

      示例

      ```
      [root@host01 ~]# cephadm shell
      ```

   d. 将 Ceph 客户端密钥环从 Ceph 监控节点复制到客户端节点：

      语法

      ```
      scp /ceph.client.ID.keyring root@CLIENT_NODE_NAME:/etc/ceph/ceph.client.ID.keyring
      ```

      将 *CLIENT_NODE_NAME* 替换为 Ceph 客户端主机名或 IP 地址。

      示例

      ```
      [ceph: root@host01 /]# scp /ceph.client.1.keyring root@client01:/etc/ceph/ceph.client.1.keyring
      ```

   e. 将 Ceph 配置文件从 Ceph 监控节点复制到客户端节点：

      语法

      ```
      scp /etc/ceph/ceph.conf root@CLIENT_NODE_NAME:/etc/ceph/ceph.conf
      ```

      将 *CLIENT_NODE_NAME* 替换为 Ceph 客户端主机名或 IP 地址。

      示例

      ```
      [ceph: root@host01 /]# scp /etc/ceph/ceph.conf root@client01:/etc/ceph/ceph.conf
      ```

   f. 在客户端节点中，为配置文件设置适当的权限：

      ```
      [root@client01 ~]# chmod 644 /etc/ceph/ceph.conf
      ```

   g. 选择 自动 或 手动 挂载.

手动挂载

2. 在客户端节点上创建挂载目录：

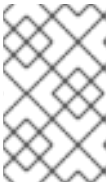
语法

```
mkdir -p MOUNT_POINT
```

示例

```
[root@client01 ~]# mkdir -p /mnt/cephfs
```

3. 挂载 Ceph 文件系统.要指定多个 Ceph 监控地址，在 **mount** 命令中使用逗号将它们分隔，指定挂载点，并设置客户端名称：



注意

自 Red Hat Ceph Storage 4.1 起，**mount.ceph** 可以直接读取密钥环文件。因此，不再需要一个 secret 文件。只需使用 **name=*CLIENT_ID*** 指定客户端 ID，**mount.ceph** 将找到正确的密钥环文件。

语法

```
mount -t ceph MONITOR-1_NAME:6789,MONITOR-2_NAME:6789,MONITOR-3_NAME:6789:/ MOUNT_POINT -o name=CLIENT_ID,fs=FILE_SYSTEM_NAME
```

示例

```
[root@client01 ~]# mount -t ceph mon1:6789,mon2:6789,mon3:6789:/ /mnt/cephfs -o name=1,fs=cephfs01
```

注意

您可以配置 DNS 服务器，以便单个主机名解析为多个 IP 地址。然后，您可以将该单一主机名与 **mount** 命令配合使用，而不必提供逗号分隔的列表。

注意

您还可以将 monitor 主机名替换为字符串 **:/** 和 **mount.ceph** 将读取 Ceph 配置文件，以确定要连接的 monitor。

注意

您可以将 **nowsync** 选项设置为在 Red Hat Ceph Storage 集群上异步执行文件创建和删除。通过避免这些系统调用的往返延迟而不会影响一致性，这提高了某些工作负载的性能。**nowsync** 选项需要带有 Red Hat Enterprise Linux 9.0 或更高版本的内核客户端。

示例

```
[root@client01 ~]# mount -t ceph mon1:6789,mon2:6789,mon3:6789:/
/mnt/cephfs -o nowsync,name=1,fs=cephfs01
```

4. 验证文件系统是否已成功挂载：

语法

```
stat -f MOUNT_POINT
```

示例

```
[root@client01 ~]# stat -f /mnt/cephfs
```

自动挂载

2. 在客户端主机上，创建用于挂载 Ceph 文件系统的新目录。

语法

```
mkdir -p MOUNT_POINT
```

示例

```
[root@client01 ~]# mkdir -p /mnt/cephfs
```

3. 编辑 **/etc/fstab** 文件，如下所示：

语法

```
#DEVICE                                PATH          TYPE    OPTIONS
MON_0_HOST:PORT,                       MOUNT_POINT   ceph    name=CLIENT_ID,
MON_1_HOST:PORT,
ceph.client_mountpoint=/VOL/SUB_VOL_GROUP/SUB_VOL/UID_SUB_VOL,
fs=FILE_SYSTEM_NAME,
MON_2_HOST:PORT:/q[_VOL_]/SUB_VOL/UID_SUB_VOL,
[ADDITIONAL_OPTIONS]
```

第一列 设置 Ceph monitor 主机名和端口号。

第二列 设置挂载点

第三列为 CephFS 设置文件系统类型，本例中为 **ceph**。

第四列 设置各种选项，如分别使用 **name** 和 **secretfile** 选项的用户名和机密文件。您还可以使用 **ceph.client_mountpoint** 选项设置特定的卷、子卷组和子卷。

设置 **_netdev** 选项，以确保在网络子系统启动后挂载文件系统，以防止挂起和网络问题。如果您不需要访问时间信息，则设置 **noatime** 选项可提高性能。

将**第五个和第六个列**设为零。

示例

```
#DEVICE        PATH                TYPE    OPTIONS         DUMP  FSCK
mon1:6789,     /mnt/cephfs           ceph    name=1,           0     0
mon2:6789,
ceph.client_mountpoint=/my_vol/my_sub_vol_group/my_sub_vol/0,
mon3:6789:/                                  fs=cephfs01,
                                    _netdev,noatime
```

Ceph 文件系统将挂载到下一次系统启动时。

注意

自 Red Hat Ceph Storage 4.1 起，**mount.ceph** 可以直接读取密钥环文件。因此，不再需要一个 secret 文件。只需使用 **name=*CLIENT_ID*** 指定客户端 ID，**mount.ceph** 将找到正确的密钥环文件。

注意

您还可以将 monitor 主机名替换为字符串 **:/** 和 **mount.ceph** 将读取 Ceph 配置文件，以确定要连接的 monitor。

其它资源

- 请参阅 **mount(8)手册**页。
- 有关创建 *Ceph 用户* 的更多详细信息，请参阅 *Red Hat Ceph Storage Administration Guide* 中的 Ceph 用户管理一章。
- 详情请参阅 *Red Hat Ceph Storage File System Guide* 中的 *Creating a Ceph File System* 部分。

3.6. 将 CEPH 文件系统挂载为 FUSE 客户端

您可以将 Ceph 文件系统 (CephFS) 作为文件系统挂载到 User Space (FUSE) 客户端中，也可以手动在系统引导时自动挂载。

先决条件

- 对基于 Linux 的客户端节点的根级别访问权限.
- Ceph 监控节点的根级别访问权限.
- 现有的 Ceph 文件系统.

流程

1. 配置客户端节点，以使用 Ceph 存储群集。

   a. 启用 Red Hat Ceph Storage 7 Tools 存储库：

      Red Hat Enterprise Linux 8

      ```
      [root@client01 ~]# subscription-manager repos --enable=6-tools-for-rhel-8-x86_64-rpms
      ```

      Red Hat Enterprise Linux 9

      ```
      [root@client01 ~]# subscription-manager repos --enable=6-tools-for-rhel-9-x86_64-rpms
      ```

   b. 安装 **ceph-fuse** 软件包：

      ```
      [root@client01 ~]# dnf install ceph-fuse
      ```

   c. 在监控节点上登录到 Cephadm shell：

      示例

      ```
      [root@host01 ~]# cephadm shell
      ```

   d. 将 Ceph 客户端密钥环从 Ceph 监控节点复制到客户端节点：

      语法

      ```
      scp /ceph.client.ID.keyring root@CLIENT_NODE_NAME:/etc/ceph/ceph.client.ID.keyring
      ```

      将 *CLIENT_NODE_NAME* 替换为 Ceph 客户端主机名或 IP 地址。

      示例

      ```
      [ceph: root@host01 /]# scp /ceph.client.1.keyring
      root@client01:/etc/ceph/ceph.client.1.keyring
      ```

   e. 将 Ceph 配置文件从 Ceph 监控节点复制到客户端节点：

      语法

      ```
      scp /etc/ceph/ceph.conf root@CLIENT_NODE_NAME:/etc/ceph/ceph.conf
      ```

      将 *CLIENT_NODE_NAME* 替换为 Ceph 客户端主机名或 IP 地址。

      示例

      ```
      [ceph: root@host01 /]# scp /etc/ceph/ceph.conf root@client01:/etc/ceph/ceph.conf
      ```

   f. 在客户端节点中，为配置文件设置适当的权限：

      ```
      [root@client01 ~]# chmod 644 /etc/ceph/ceph.conf
      ```

g. 选择 自动 或 手动 挂载.

手动挂载

2. 在客户端节点上，为挂载点创建一个目录：

语法

```
mkdir PATH_TO_MOUNT_POINT
```

示例

```
[root@client01 ~]# mkdir /mnt/mycephfs
```

注意

如果您将 **path** 选项与 MDS 功能一起使用，则挂载点必须在 **pat** 中指定的范围内。

3. 使用 **ceph-fuse** 实用程序挂载 Ceph 文件系统。

语法

```
ceph-fuse -n client.CLIENT_ID --client_fs FILE_SYSTEM_NAME MOUNT_POINT
```

示例

```
[root@client01 ~]# ceph-fuse -n client.1 --client_fs cephfs01 /mnt/mycephfs
```

注意

如果您不使用用户密钥环的默认名称和位置，即 **/etc/ceph/ceph.client.*CLIENT_ID*.keyring**，则使用 **--keyring** 选项指定用户密钥环的路径，例如：

示例

```
[root@client01 ~]# ceph-fuse -n client.1 --keyring=/etc/ceph/client.1.keyring
/mnt/mycephfs
```

注意

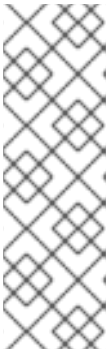
使用 **-r** 选项指示客户端将该路径视为其根路径：

语法

```
ceph-fuse -n client.CLIENT_ID MOUNT_POINT -r PATH
```

示例

```
[root@client01 ~]# ceph-fuse -n client.1 /mnt/cephfs -r /home/cephfs
```



注意

如果要自动重新连接被驱除的 Ceph 客户端，请添加 **--client_reconnect_stale=true** 选项。

示例

```
[root@client01 ~]# ceph-fuse -n client.1 /mnt/cephfs --
client_reconnect_stale=true
```

4. 验证文件系统是否已成功挂载：

语法

```
stat -f MOUNT_POINT
```

示例

```
[root@client01 ~]# stat -f /mnt/cephfs
```

自动挂载

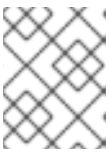
2. 在客户端节点上，为挂载点创建一个目录：

语法

```
mkdir PATH_TO_MOUNT_POINT
```

示例

```
[root@client01 ~]# mkdir /mnt/mycephfs
```



注意

如果您将 **path** 选项与 MDS 功能一起使用，则挂载点必须在 **pat** 中指定的范围内。

3. 编辑 **/etc/fstab** 文件，如下所示：

语法

```
#DEVICE                PATH          TYPE          OPTIONS                  DUMP  FSCK
HOST_NAME:PORT,    MOUNT_POINT fuse.ceph     ceph.id=CLIENT_ID,       0     0
HOST_NAME:PORT,
ceph.client_mountpoint=/VOL/SUB_VOL_GROUP/SUB_VOL/UID_SUB_VOL,
HOST_NAME:PORT:/
ceph.client_fs=FILE_SYSTEM_NAME,ceph.name=USERNAME,ceph.keyring=/etc/ceph/KEY
RING_FILE,
                                                [ADDITIONAL_OPTIONS]
```

第一列 设置 Ceph monitor 主机名和端口号。

第二列 设置挂载点

第三列为 CephFS 设置文件系统类型，本例中为 **fuse.ceph**。

第四列设置各种选项，如使用 **ceph.name** 和 **ceph.keyring** 选项的用户名和密钥环。您还可以使用 **ceph.client_mountpoint** 选项设置特定的卷、子卷组和子卷。要指定可访问的 Ceph 文件系统，请使用 **ceph.client_fs** 选项。设置 **_netdev** 选项，以确保在网络子系统启动后挂载文件系统，以防止挂起和网络问题。如果您不需要访问时间信息，则设置 **noatime** 选项可提高性能。如果要在驱除后自动重新连接，请设置 **client_reconnect_stale=true** 选项。

将**第五个和第六个列**设为零。

示例

```
#DEVICE        PATH            TYPE        OPTIONS        DUMP  FSCK
mon1:6789,     /mnt/mycephfs       fuse.ceph    ceph.id=1,        0     0
mon2:6789,
ceph.client_mountpoint=/my_vol/my_sub_vol_group/my_sub_vol/0,
mon3:6789:/
ceph.client_fs=cephfs01,ceph.name=client.1,ceph.keyring=/etc/ceph/client1.keyring,
                                    _netdev,defaults
```

Ceph 文件系统将挂载到下一次系统启动时。

其它资源

- **ceph-fuse(8)手册**页。
- 如需了解有关创建 Ceph 用户的更多详细信息，请参见 *Red Hat Ceph Storage Administration Guide* 中的 *Ceph user management* 部分。
- 详情请参阅 *Red Hat Ceph Storage File System Guide* 中的 *Creating a Ceph File System* 部分。

其它资源

- 请参阅 第 2.5 节 “使用 Ceph Orchestrator 管理 MDS 服务” 来安装 Ceph 元数据服务器。
- 详情请查看 第 3.2 节 “创建 Ceph 文件系统”。
- 详情请查看 第 3.4 节 “为 Ceph 文件系统创建客户端用户”。
- 详情请查看 第 3.5 节 “将 Ceph 文件系统挂载为内核客户端”。

- 详情请查看 第 3.6 节 “将 Ceph 文件系统挂载为 FUSE 客户端” 。
- 有关配置 CephFS 元数据服务器守护进程的详细信息，请参阅 第 2 章 *Ceph 文件系统元数据服务器*。

第 4 章 管理 CEPH 文件系统卷、子卷组和子卷

作为存储管理员，您可以使用红帽的 Ceph Container Storage Interface (CSI) 管理 Ceph 文件系统 (CephFS) 导出。这也允许您通过具有可与之交互的通用命令行界面来使用 OpenStack 文件系统服务 (Manila) 等其他服务。Ceph 管理器守护进程 (**ceph-mgr**) 的 **volumes** 模块实施导出 Ceph 文件系统 (CephFS) 的功能。

Ceph Manager volumes 模块实施以下文件系统导出抽象：

- CephFS 卷
- CephFS 子卷组
- CephFS 子卷

4.1. CEPH 文件系统卷

作为存储管理员，您可以创建、列出和删除 Ceph 文件系统 (CephFS) 卷。CephFS 卷是 Ceph 文件系统的抽象。

本节描述了如何：

- 创建 Ceph 文件系统卷。
- 列出 Ceph 文件系统卷。
- 查看有关 Ceph 文件系统卷的信息。
- 移除 Ceph 文件系统卷。

4.1.1. 创建 Ceph 文件系统卷

Ceph 编排器是 Ceph 管理器的一个模块，可为 Ceph 文件系统(CephFS)创建元数据服务器(MDS)。本节介绍如何创建 CephFS 卷。

注意

这将创建 Ceph 文件系统，以及数据和元数据池。

先决条件

- 部署的 Ceph 文件系统正常工作的 Red Hat Ceph Storage 存储群集。
- 至少对 Ceph 监控器具有读取访问权限。
- Ceph 管理器节点上的读写功能。

流程

- 在监控节点上创建 CephFS 卷：

  语法

  ```
  ceph fs volume create VOLUME_NAME
  ```

示例

```
[ceph: root@host01 /]# ceph fs volume create cephfs
```

4.1.2. 列出 Ceph 文件系统卷

本节介绍列出 Ceph 文件系统 (CephFS) 卷的步骤。

先决条件

- 部署的 Ceph 文件系统正常工作的 Red Hat Ceph Storage 存储群集。
- 至少对 Ceph 监控器具有读取访问权限。
- Ceph 管理器节点上的读写功能。
- CephFS 卷。

流程

- 列出 CephFS 卷：

  示例

  ```
  [ceph: root@host01 /]# ceph fs volume ls
  ```

4.1.3. 查看有关 Ceph 文件系统卷的信息

您可以列出 Ceph 文件系统(CephFS)卷的基本详情，如 CephFS 卷的数据和元数据池的属性、待处理子卷删除计数，等等。

先决条件

- 部署的 Ceph 文件系统正常工作的 Red Hat Ceph Storage 存储群集。
- 至少对 Ceph 监控器具有读取访问权限。
- Ceph 管理器节点上的读写功能。
- 创建了一个 CephFS 卷。

流程

- 查看 CephFS 卷的信息：

  语法

  ```
  ceph fs volume info VOLUME_NAME
  ```

  示例

  ```
  [ceph: root@host01 /]# ceph fs volume info cephfs
  {
  ```

```
    "mon_addrs": [
        "192.168.1.7:40977",
    ],
    "pending_subvolume_deletions": 0,
    "pools": {
        "data": [
            {
                "avail": 106288709632,
                "name": "cephfs.cephfs.data",
                "used": 4096
            }
        ],
        "metadata": [
            {
                "avail": 106288709632,
                "name": "cephfs.cephfs.meta",
                "used": 155648
            }
        ]
    },
    "used_size": 0
}
```


ceph fs volume info 命令的输出包括：

- **mon_addrs** ：监控地址列表。
- **pending_subvolume_deletions** ：子卷数量待处理删除。
- **池** ：数据和元数据池的属性.
  - **avail** ：可用的空间量（以字节为单位）。
  - **名称** ：池的名称。
  - **使用的** ：以字节为单位消耗的存储量。
- **used_size** ：当前使用的 CephFS 卷的大小（以字节为单位）。

4.1.4. 删除 Ceph 文件系统卷

Ceph 编排器是 Ceph 管理器的一个模块，用于删除 Ceph 文件系统(CephFS)的元数据服务器(MDS)。本节介绍如何删除 Ceph 文件系统 (CephFS) 卷。

先决条件

- 部署的 Ceph 文件系统正常工作的 Red Hat Ceph Storage 存储群集。
- 至少对 Ceph 监控器具有读取访问权限。
- Ceph 管理器节点上的读写功能。
- CephFS 卷。

流程

1. 如果 **mon_allow_pool_delete** 选项还没有被设置为 **true**，则先将它设置为 **true**，然后删除 CephFS 卷：

   示例

   ```
   [ceph: root@host01 /]# ceph config set mon mon_allow_pool_delete true
   ```

2. 移除 CephFS 卷：

   语法

   ```
   ceph fs volume rm VOLUME_NAME [--yes-i-really-mean-it]
   ```

   示例

   ```
   [ceph: root@host01 /]# ceph fs volume rm cephfs --yes-i-really-mean-it
   ```

4.2. CEPH 文件系统子卷组

作为存储管理员，您可以创建、列出、获取绝对路径，以及删除 Ceph 文件系统 (CephFS) 子卷组。CephFS 子卷组是目录级别的抽象，对一组子卷的影响策略（如文件布局）。

从 Red Hat Ceph Storage 5.0 开始，不支持 subvolume 组群快照功能。您只能列出并删除这些子卷组的现有快照。

本节描述了如何：

- 创建文件系统子卷组。
- 在文件系统子卷组上设置和管理配额。
- 列出文件系统子卷组。
- 获取文件系统子卷组的绝对路径。
- 列出文件系统子卷组的快照。
- 删除文件系统子卷组的快照。
- 删除文件系统子卷组。

4.2.1. 创建文件系统子卷组

这部分论述了如何创建 Ceph 文件系统 (CephFS) 子卷组。

注意

在创建子卷组时，您可以在八进制数中指定其数据池布局、uid、gid 和文件模式。默认情况下，使用八进制文件模式 '755'、uid '0'、gid '0' 和其父目录的数据池布局创建子卷组。

注意

请参阅在文件系统子卷组中设置和管理配额，以便在创建子卷组时设置配额。

先决条件

- 部署的 Ceph 文件系统正常工作的 Red Hat Ceph Storage 存储群集。
- 至少对 Ceph 监控器具有读取访问权限。
- Ceph 管理器节点上的读写功能。

流程

- 创建 CephFS 子卷组：

  语法

  ```
  ceph fs subvolumegroup create VOLUME_NAME GROUP_NAME [--pool_layout DATA_POOL_NAME --uid UID --gid GID --mode OCTAL_MODE]
  ```

  示例

  ```
  [ceph: root@host01 /]# ceph fs subvolumegroup create cephfs subgroup0
  ```

  即使子卷组已存在，命令也会成功。

4.2.2. 在文件系统子卷组中设置和管理配额

本节论述了如何在 Ceph 文件系统(CephFS)子卷组上设置和管理配额。

先决条件

- 部署的 Ceph 文件系统正常工作的 Red Hat Ceph Storage 存储群集。
- 至少对 Ceph 监控器具有读取访问权限。
- Ceph 管理器节点上的读写功能。

流程

1. 通过提供大小（以字节为单位）在创建子卷组时设置配额：

   语法

   ```
   ceph fs subvolumegroup create VOLUME_NAME GROUP_NAME [--size SIZE_IN_BYTES] [--pool_layout DATA_POOL_NAME] [--uid UID] [--gid GID] [--mode OCTAL_MODE]
   ```

   示例

   ```
   [ceph: root@host01 /]# ceph fs subvolumegroup create cephfs subvolgroup_2 10737418240
   ```

2. 重新定义子卷组大小：

   语法

   ```
   ceph fs subvolumegroup resize VOLUME_NAME GROUP_NAME new_size [--no_shrink]
   ```

示例

```
[ceph: root@host01 /]# ceph fs subvolumegroup resize cephfs subvolgroup_2 20737418240
[
    {
        "bytes_used": 10768679044
    },
    {
        "bytes_quota": 20737418240
    },
    {
        "bytes_pcent": "51.93"
    }
]
```

3. 获取子卷组的元数据：

语法

```
ceph fs subvolumegroup info VOLUME_NAME GROUP_NAME
```

示例

```
[ceph: root@host01 /]# ceph fs subvolumegroup info cephfs subvolgroup_2
{
    "atime": "2022-10-05 18:00:39",
    "bytes_pcent": "51.85",
    "bytes_quota": 20768679043,
    "bytes_used": 10768679044,
    "created_at": "2022-10-05 18:00:39",
    "ctime": "2022-10-05 18:21:26",
    "data_pool": "cephfs.cephfs.data",
    "gid": 0,
    "mode": 16877,
    "mon_addrs": [
        "60.221.178.236:1221",
        "205.64.75.112:1221",
        "20.209.241.242:1221"
    ],
    "mtime": "2022-10-05 18:01:25",
    "uid": 0
}
```

4.2.3. 列出文件系统子卷组

本节介绍列出 Ceph 文件系统 (CephFS) 子卷组的步骤。

先决条件

- 部署的 Ceph 文件系统正常工作的 Red Hat Ceph Storage 存储群集。
- 至少对 Ceph 监控器具有读取访问权限。
- Ceph 管理器节点上的读写功能。

- CephFS 子卷组。

流程

- 列出 CephFS 子卷组：

  语法

  ```
  ceph fs subvolumegroup ls VOLUME_NAME
  ```

  示例

  ```
  [ceph: root@host01 /]# ceph fs subvolumegroup ls cephfs
  ```

4.2.4. 获取文件系统子卷组的绝对路径

本节介绍如何获取 Ceph 文件系统 (CephFS) 子卷组的绝对路径。

先决条件

- 部署的 Ceph 文件系统正常工作的 Red Hat Ceph Storage 存储群集。
- 至少对 Ceph 监控器具有读取访问权限。
- Ceph 管理器节点上的读写功能。
- CephFS 子卷组。

流程

- 获取 CephFS 子卷组的绝对路径：

  语法

  ```
  ceph fs subvolumegroup getpath VOLUME_NAME GROUP_NAME
  ```

  示例

  ```
  [ceph: root@host01 /]# ceph fs subvolumegroup getpath cephfs subgroup0
  ```

4.2.5. 列出文件系统子卷组的快照

本节提供列出 Ceph 文件系统 (CephFS) 子卷组快照的步骤。

先决条件

- 部署的 Ceph 文件系统正常工作的 Red Hat Ceph Storage 存储群集。
- 至少对 Ceph 监控器具有读取访问权限。
- Ceph 管理器节点上的读写功能。

- CephFS 子卷组。
- 子卷组的快照。

流程

- 列出 CephFS 子卷组的快照：

  语法

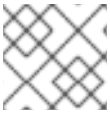
  ```
  ceph fs subvolumegroup snapshot ls VOLUME_NAME GROUP_NAME
  ```

  示例

  ```
  [ceph: root@host01 /]# ceph fs subvolumegroup snapshot ls cephfs subgroup0
  ```

4.2.6. 删除文件系统子卷组的快照

本节提供删除 Ceph 文件系统 (CephFS) 子卷组快照的步骤。



注意

使用 **--force** 标志时，命令可以成功，否则如果快照不存在，则会失败。

先决条件

- 部署的 Ceph 文件系统正常工作的 Red Hat Ceph Storage 存储群集。
- 至少对 Ceph 监控器具有读取访问权限。
- Ceph 管理器节点上的读写功能。
- Ceph 文件系统卷.
- 子卷组的快照。

流程

- 移除 CephFS 子卷组的快照：

  语法

  ```
  ceph fs subvolumegroup snapshot rm VOLUME_NAME GROUP_NAME SNAP_NAME [--force]
  ```

  示例

  ```
  [ceph: root@host01 /]# ceph fs subvolumegroup snapshot rm cephfs subgroup0 snap0 --force
  ```

4.2.7. 删除文件系统子卷组

本节介绍如何删除 Ceph 文件系统 (CephFS) 子卷组。

注意

如果子卷组未为空或不存在，则移除子卷组会失败。**--force** 标志允许删除不存在的子卷组。

先决条件

- 部署的 Ceph 文件系统正常工作的 Red Hat Ceph Storage 存储群集。
- 至少对 Ceph 监控器具有读取访问权限。
- Ceph 管理器节点上的读写功能。
- CephFS 子卷组。

流程

- 删除 CephFS 子卷组：

  语法

  ```
  ceph fs subvolumegroup rm VOLUME_NAME GROUP_NAME [--force]
  ```

  示例

  ```
  [ceph: root@host01 /]# ceph fs subvolumegroup rm cephfs subgroup0 --force
  ```

4.3. CEPH 文件系统子卷

作为存储管理员，您可以创建、列出、获取绝对路径、获取元数据，以及移除 Ceph 文件系统 (CephFS) 子卷。此外，您也可以创建、列出和删除这些子卷的快照。CephFS 子卷是独立 Ceph 文件系统目录树的抽象。

本节描述了如何：

- 创建文件系统子卷。
- 列出文件系统子卷。
- 重新定义文件系统子卷大小。
- 获取文件系统子卷的绝对路径。
- 获取文件系统子卷的元数据。
- 为文件系统子卷创建快照。
- 从快照克隆子卷。
- 列出文件系统子卷的快照。
- 获取文件系统子卷的快照元数据。

- 删除文件系统子卷。
- 删除文件系统子卷的快照。

4.3.1. 创建文件系统子卷

这部分论述了如何创建 Ceph 文件系统 (CephFS) 子卷。

注意

在创建子卷时，您可以指定其子卷组、数据池布局、uid、gid、八进制数字文件模式和大小（以字节为单位）。可以通过指定 **--namespace-isolated** 选项，在单独的 RADOS 命名空间中创建子卷。默认情况下，子卷在默认子卷组中创建，使用八进制文件模式 '755'、子卷组的 uid、子卷组 gid、其父目录的数据池布局和无大小限制。

先决条件

- 部署的 Ceph 文件系统正常工作的 Red Hat Ceph Storage 存储群集。
- 至少对 Ceph 监控器具有读取访问权限。
- Ceph 管理器节点上的读写功能。

流程

- 创建 CephFS 子卷：

  语法

  ceph fs subvolume create *VOLUME_NAME SUBVOLUME_NAME* [--size *SIZE_IN_BYTES* --group_name *SUBVOLUME_GROUP_NAME* --pool_layout *DATA_POOL_NAME --uid _UID* --gid *GID* --mode *OCTAL_MODE*] [--namespace-isolated]

  示例

  ```
  [root@mon ~]# ceph fs subvolume create cephfs sub0 --group_name subgroup0 --namespace-isolated
  ```

  即使子卷已存在，命令也会成功。

4.3.2. 列出文件系统子卷

本节介绍列出 Ceph 文件系统 (CephFS) 子卷的步骤。

先决条件

- 部署的 Ceph 文件系统正常工作的 Red Hat Ceph Storage 存储群集。
- 至少对 Ceph 监控器具有读取访问权限。
- Ceph 管理器节点上的读写功能。
- CephFS 子卷。

流程

- 列出 CephFS 子卷：

  语法

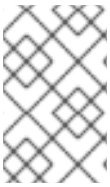
  ```
  ceph fs subvolume ls VOLUME_NAME [--group_name SUBVOLUME_GROUP_NAME]
  ```

  示例

  ```
  [root@mon ~]# ceph fs subvolume ls cephfs --group_name subgroup0
  ```

4.3.3. 重新定义文件系统子卷大小

本节介绍调整 Ceph 文件系统 (CephFS) 子卷大小的步骤。



注意

ceph fs subvolume resize 命令使用 **new_size** 指定的大小来调整子卷配额的大小。**--no_shrink** 标志可防止子卷的大小被减少到当前使用的子卷大小。通过将 **new_size** 设置为 **inf** 或 **infinite** 来将子卷重新设置为一个无限的大小。

先决条件

- 部署的 Ceph 文件系统正常工作的 Red Hat Ceph Storage 存储群集。
- 至少对 Ceph 监控器具有读取访问权限。
- Ceph 管理器节点上的读写功能。
- CephFS 子卷。

流程

- 重新定义 CephFS 子卷大小：

  语法

  ```
  ceph fs subvolume resize VOLUME_NAME SUBVOLUME_NAME NEW_SIZE [--group_name SUBVOLUME_GROUP_NAME] [--no_shrink]
  ```

  示例

  ```
  [root@mon ~]# ceph fs subvolume resize cephfs sub0 1024000000 --group_name subgroup0 --no_shrink
  ```

4.3.4. 获取文件系统子卷的绝对路径

本节介绍如何获取 Ceph 文件系统 (CephFS) 子卷的绝对路径。

先决条件

- 部署的 Ceph 文件系统正常工作的 Red Hat Ceph Storage 存储群集。
- 至少对 Ceph 监控器具有读取访问权限。
- Ceph 管理器节点上的读写功能。
- CephFS 子卷。

流程

- 获取 CephFS 子卷的绝对路径：

  语法

  ```
  ceph fs subvolume getpath VOLUME_NAME SUBVOLUME_NAME [--group_name _SUBVOLUME_GROUP_NAME]
  ```

  示例

  ```
  [root@mon ~]# ceph fs subvolume getpath cephfs sub0 --group_name subgroup0
  ```

4.3.5. 获取文件系统子卷的元数据

本节介绍如何获取 Ceph 文件系统 (CephFS) 子卷的元数据。

先决条件

- 部署的 Ceph 文件系统正常工作的 Red Hat Ceph Storage 存储群集。
- 至少对 Ceph 监控器具有读取访问权限。
- Ceph 管理器节点上的读写功能。
- CephFS 子卷。

流程

- 获取 CephFS 子卷的元数据：

  语法

  ```
  ceph fs subvolume info VOLUME_NAME SUBVOLUME_NAME [--group_name SUBVOLUME_GROUP_NAME]
  ```

  示例

  ```
  [root@mon ~]# ceph fs subvolume info cephfs sub0 --group_name subgroup0
  ```

  输出示例

  ```
  # ceph fs subvolume info cephfs sub0
  {
      "atime": "2023-07-14 08:52:46",
  ```

```
    "bytes_pcent": "0.00",
    "bytes_quota": 1024000000,
    "bytes_used": 0,
    "created_at": "2023-07-14 08:52:46",
    "ctime": "2023-07-14 08:53:54",
    "data_pool": "cephfs.cephfs.data",
    "features": [
        "snapshot-clone",
        "snapshot-autoprotect",
        "snapshot-retention"
    ],
    "flavor": "2",
    "gid": 0,
    "mode": 16877,
    "mon_addrs": [
        "10.0.208.172:6789",
        "10.0.211.197:6789",
        "10.0.209.212:6789"
    ],
    "mtime": "2023-07-14 08:52:46",
    "path": "/volumes/_nogroup/sub0/834c5cbc-f5db-4481-80a3-aca92ff0e7f3",
    "pool_namespace": "",
    "state": "complete",
    "type": "subvolume",
    "uid": 0
}
```


输出格式为 JSON，包含以下字段：

- **atime** ：访问子卷路径的时间，格式为 "YYYY-MM-DD HH:MM:SS"。
- **bytes_pcent** ：如果设置了配额，则以百分比为单位使用的配额，否则会显示"undefined"。
- **bytes_quota** ：如果设置了配额，则配额大小以字节为单位，否则会显示"infinite"。
- **bytes_used** ：子卷的当前使用大小（以字节为单位）。
- **created_at** ：创建子卷的时间，格式为 "YYYY-MM-DD HH:MM:SS"。
- **ctime** ：更改子卷路径的时间，格式为 "YYYY-MM-DD HH:MM:SS"。
- **data_pool** ：子卷所属的数据池。
- **features**： 子卷支持的功能，如 snapshot-clone"、"snapshot-autoprotect" 或 "snapshot-retention"。
- **类别** ：子卷版本，可以是 **1 (** 版本 1)或版本 **2**。
- **GID** ：子卷路径的组 ID。
- **mode** ：子卷路径的模式。
- **mon_addrs** ：监控地址列表。
- **mtime** ：修改子卷路径的时间，格式为"YYYY-MM-DD HH:MM:SS"。
- **path** ：子卷的绝对路径。

- **pool_namespace** ：子卷的 RADOS 命名空间。
- **State** ：子卷的当前状态，如 "complete" 或 "snapshot-retained"。
- **类型** ：子卷类型，指示它是克隆还是子卷。
- **UID** ：子卷路径的用户 ID。

4.3.6. 创建文件系统子卷的快照

本节介绍如何创建 Ceph 文件系统 (CephFS) 子卷的快照。

先决条件

- 部署的 Ceph 文件系统正常工作的 Red Hat Ceph Storage 存储群集。
- 至少对 Ceph 监控器具有读取访问权限。
- Ceph 管理器节点上的读写功能。
- CephFS 子卷。
- 除了读取 (**r**) 和写入 (**w**) 功能外，客户端还需要文件系统的目录路径上的 **s** 标志。

流程

1. 验证目录中是否设置了 **s** 标记：

   语法

   ```
   ceph auth get CLIENT_NAME
   ```

   示例

   ```
   [root@mon ~]# ceph auth get client.0
   [client.0]
       key = AQAz7EVWygILFRAAdIcuJ12opU/JKyfFmxhuaw==
       caps mds = "allow rw, allow rws path=/bar" (1)
       caps mon = "allow r"
       caps osd = "allow rw tag cephfs data=cephfs_a" (2)
   ```

   (1) (2) 在示例中，**client.0** 可以在文件系统 **cephfs_a** 的 **bar** 目录中创建或删除快照。

2. 创建 Ceph 文件系统子卷的快照：

   语法

   ```
   ceph fs subvolume snapshot create VOLUME_NAME SUBVOLUME_NAME SNAP_NAME [--group_name GROUP_NAME]
   ```

   示例

```
[root@mon ~]# ceph fs subvolume snapshot create cephfs sub0 snap0 --group_name
subgroup0
```

4.3.7. 从快照克隆子卷

可以通过克隆子卷快照来创建子卷。这是一个异步操作，涉及将快照中的数据复制到子卷。

注意

对于非常大的数据集，克隆效率较低。

先决条件

- 部署的 Ceph 文件系统正常工作的 Red Hat Ceph Storage 存储群集。
- 至少对 Ceph 监控器具有读取访问权限。
- Ceph 管理器节点上的读写功能。
- 要创建或删除快照，除了读写功能外，客户端还需要文件系统中目录路径上的 **s** 标志。

语法

```
CLIENT_NAME
   key = AQAz7EVWygILFRAAdIcuJ12opU/JKyfFmxhuaw==
   caps mds = allow rw, allow rws path=DIRECTORY_PATH
   caps mon = allow r
   caps osd = allow rw tag cephfs data=DIRECTORY_NAME
```

在以下示例中，**client.0** 可以在文件系统 **cephfs_a** 的 **bar** 目录中创建或删除快照。

示例

```
[client.0]
   key = AQAz7EVWygILFRAAdIcuJ12opU/JKyfFmxhuaw==
   caps mds = "allow rw, allow rws path=/bar"
   caps mon = "allow r"
   caps osd = "allow rw tag cephfs data=cephfs_a"
```

流程

1. 创建 Ceph 文件系统 (CephFS) 卷：

语法

```
ceph fs volume create VOLUME_NAME
```

示例

```
[root@mon ~]# ceph fs volume create cephfs
```

这将创建 CephFS 文件系统、其数据和元数据池。

2. 创建子卷组。默认情况下，使用八进制文件模式 '755' 创建子卷组，以及其父目录的数据池布局。

语法

```
ceph fs subvolumegroup create VOLUME_NAME GROUP_NAME [--pool_layout
DATA_POOL_NAME --uid UID --gid GID --mode OCTAL_MODE]
```

示例

```
[root@mon ~]# ceph fs subvolumegroup create cephfs subgroup0
```

3. 创建子卷。默认情况下，子卷在默认子卷组中创建，使用八进制文件模式 '755'、子卷组的 uid、子卷组 gid、其父目录的数据池布局和无大小限制。

语法

```
ceph fs subvolume create VOLUME_NAME SUBVOLUME_NAME [--size SIZE_IN_BYTES --
-group_name SUBVOLUME_GROUP_NAME --pool_layout DATA_POOL_NAME --uid _UID
--gid GID --mode OCTAL_MODE]
```

示例

```
[root@mon ~]# ceph fs subvolume create cephfs sub0 --group_name subgroup0
```

4. 创建子卷的快照：

语法

```
ceph fs subvolume snapshot create VOLUME_NAME _SUBVOLUME_NAME SNAP_NAME
[--group_name SUBVOLUME_GROUP_NAME]
```

示例

```
[root@mon ~]# ceph fs subvolume snapshot create cephfs sub0 snap0  --group_name
subgroup0
```

5. 启动克隆操作：

注意

默认情况下，克隆的子卷会在默认组中创建。

a. 如果源子卷和目标克隆位于默认组中，请运行以下命令：

语法

```
ceph fs subvolume snapshot clone VOLUME_NAME SUBVOLUME_NAME
SNAP_NAME TARGET_CLONE_NAME
```

示例

```
[root@mon ~]# ceph fs subvolume snapshot clone cephfs sub0 snap0 clone0
```

b. 如果源子卷位于非默认组中，请使用以下命令指定源子卷组：

语法

```
ceph fs subvolume snapshot clone VOLUME_NAME SUBVOLUME_NAME
SNAP_NAME TARGET_CLONE_NAME --group_name SUBVOLUME_GROUP_NAME
```

示例

```
[root@mon ~]# ceph fs subvolume snapshot clone cephfs sub0 snap0 clone0 --
group_name subgroup0
```

c. 如果目标克隆到非默认组中，请使用以下命令指定目标组：

语法

```
ceph fs subvolume snapshot clone VOLUME_NAME SUBVOLUME_NAME
SNAP_NAME TARGET_CLONE_NAME --target_group_name
SUBVOLUME_GROUP_NAME
```

示例

```
[root@mon ~]# ceph fs subvolume snapshot clone cephfs sub0 snap0 clone0 --
target_group_name subgroup1
```

6. 检查克隆操作的状态：

语法

```
ceph fs clone status VOLUME_NAME CLONE_NAME [--group_name
TARGET_GROUP_NAME]
```

示例

```
[root@mon ~]# ceph fs clone status cephfs clone0 --group_name subgroup1

{
  "status": {
    "state": "complete"
  }
}
```

其它资源

- 请参阅 *Red Hat Ceph Storage Administration Guide* 中的 *Managing Ceph users* 章节。

4.3.8. 列出文件系统子卷的快照

本节提供列出 Ceph 文件系统 (CephFS) 子卷快照的步骤。

先决条件

- 部署的 Ceph 文件系统正常工作的 Red Hat Ceph Storage 存储群集。
- 至少对 Ceph 监控器具有读取访问权限。
- Ceph 管理器节点上的读写功能。
- CephFS 子卷。
- 子卷的快照。

流程

- 列出 CephFS 子卷的快照：

  语法

  ```
  ceph fs subvolume snapshot ls VOLUME_NAME SUBVOLUME_NAME [--group_name SUBVOLUME_GROUP_NAME]
  ```

  示例

  ```
  [root@mon ~]# ceph fs subvolume snapshot ls cephfs sub0 --group_name subgroup0
  ```

4.3.9. 获取文件系统子卷快照的元数据

本节提供获取 Ceph 文件系统 (CephFS) 子卷快照元数据的步骤。

先决条件

- 部署的 CephFS 正常工作的 Red Hat Ceph Storage 集群。
- 至少对 Ceph 监控器具有读取访问权限。
- Ceph 管理器节点上的读写功能。
- CephFS 子卷。
- 子卷的快照。

流程

1. 获取 CephFS 子卷的快照元数据：

   语法

   ```
   ceph fs subvolume snapshot info VOLUME_NAME SUBVOLUME_NAME SNAP_NAME [--group_name SUBVOLUME_GROUP_NAME]
   ```

   示例

   ```
   [root@mon ~]# ceph fs subvolume snapshot info cephfs sub0 snap0 --group_name subgroup0
   ```

-

输出示例

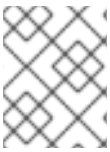
```
{
    "created_at": "2022-05-09 06:18:47.330682",
    "data_pool": "cephfs_data",
    "has_pending_clones": "no",
    "size": 0
}
```

输出格式为 JSON，包含以下字段：

- **created_at** ：创建快照的时间，格式为 "YYYY-MM-DD HH:MM:SS:ffff"。
- **data_pool** ：快照所属的数据池。
- **has_pending_clones**: "yes" 如果快照克隆正在进行中，否则为"no"。
- **size** ：快照大小，以字节为单位.

4.3.10. 删除文件系统子卷

本节介绍删除 Ceph 文件系统 (CephFS) 子卷的步骤。



注意

ceph fs subvolume rm 命令会在两个步骤中删除子卷及其内容。首先，它会将子卷移到回收文件夹中，然后异步清除其内容。

可以使用 **--retain-snapshots** 选项删除子卷的现有快照。如果保留快照，则所有不涉及保留快照的操作的子卷将被视为空。保留的快照可用作克隆源来重新创建子卷，或克隆到较新的子卷。

先决条件

- 部署的 Ceph 文件系统正常工作的 Red Hat Ceph Storage 存储群集。
- 至少对 Ceph 监控器具有读取访问权限。
- Ceph 管理器节点上的读写功能。
- CephFS 子卷。

流程

1. 删除 CephFS 子卷：

语法

```
ceph fs subvolume rm VOLUME_NAME SUBVOLUME_NAME [--group_name SUBVOLUME_GROUP_NAME] [--force] [--retain-snapshots]
```

示例

```
[root@mon ~]# ceph fs subvolume rm cephfs sub0 --group_name subgroup0 --retain-
snapshots
```

2. 从保留的快照重新创建子卷：

语法

```
ceph fs subvolume snapshot clone VOLUME_NAME DELETED_SUBVOLUME
RETAINED_SNAPSHOT NEW_SUBVOLUME --group_name
SUBVOLUME_GROUP_NAME --target_group_name
SUBVOLUME_TARGET_GROUP_NAME
```

- *NEW_SUBVOLUME* 可以是之前删除的同一子卷，也可以克隆到新子卷中。

示例

```
[root@mon ~]# ceph fs subvolume snapshot clone cephfs sub0 snap0 sub1 --group_name
subgroup0 --target_group_name subgroup0
```

4.3.11. 删除文件系统子卷的快照

本节提供删除 Ceph 文件系统 (CephFS) 子卷组快照的步骤。

注意

使用 **--force** 标志时，命令可以成功，否则如果快照不存在，则会失败。

先决条件

- 部署的 Ceph 文件系统正常工作的 Red Hat Ceph Storage 存储群集。
- 至少对 Ceph 监控器具有读取访问权限。
- Ceph 管理器节点上的读写功能。
- Ceph 文件系统卷.
- 子卷组的快照。

流程

- 移除 CephFS 子卷的快照：

语法

```
ceph fs subvolume snapshot rm VOLUME_NAME SUBVOLUME_NAME SNAP_NAME [--
group_name GROUP_NAME --force]
```

示例

```
[root@mon ~]# ceph fs subvolume snapshot rm cephfs sub0 snap0 --group_name
subgroup0 --force
```

其它资源

- 请参阅 *Red Hat Ceph Storage Administration Guide* 中的 *Managing Ceph users* 章节。

4.4. CEPH 文件系统子卷的元数据信息

作为存储管理员，您可以设置、获取、列出和删除 Ceph 文件系统(CephFS)子卷的元数据信息。

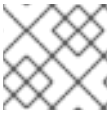
自定义元数据供用户在子卷中存储其元数据。用户可以将类似 **xattr** 的键值对存储在 Ceph 文件系统中。

本节描述了如何：

- 在文件系统子卷中设置自定义元数据
- 在文件系统子卷中获取自定义元数据
- 列出文件系统子卷上的自定义元数据
- 从文件系统子卷中删除自定义元数据

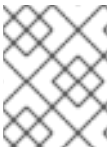
4.4.1. 在文件系统子卷中设置自定义元数据

您可以将文件系统子卷上的自定义元数据设置为键值对。



注意

如果 **key_name** 已存在，则旧值将被新值替代。



注意

KEY_NAME 和 ***VALUE*** 应该是在 python 的 **string.printable** 中指定的 ASCII 字符字符串。***KEY_NAME*** 是区分大小写的，始终存储在小写中。

重要

当快照子卷时，子卷上的自定义元数据不会被保留，因此在克隆子卷快照时也不会保留。

先决条件

- 一个正在运行的 Red Hat Ceph Storage 集群。
- 已创建 Ceph 文件系统(CephFS)、CephFS 卷、子卷组和子卷。

流程

1. 在 CephFS 子卷上设置元数据：

   语法

   ```
   ceph fs subvolume metadata set VOLUME_NAME SUBVOLUME_NAME KEY_NAME VALUE [--group_name SUBVOLUME_GROUP_NAME]
   ```

   示例

   -

```
[ceph: root@host01 /]# ceph fs subvolume metadata set cephfs sub0 test_meta cluster --
group_name subgroup0
```

2. 可选：使用 ***KEY_NAME*** 中的空格设置自定义元数据：

   示例

   ```
   [ceph: root@host01 /]# ceph fs subvolume metadata set cephfs sub0 "test meta" cluster --
   group_name subgroup0
   ```

   这会为 *VALUE* **集群** 创建另一个带有 ***KEY_NAME*** 作为 **test meta** 的元数据。

3. 可选：您还可以使用不同的值设置相同的元数据：

   示例

   ```
   [ceph: root@host01 /]# ceph fs subvolume metadata set cephfs sub0 "test_meta" cluster2 --
   group_name subgroup0
   ```

4.4.2. 在文件系统子卷中获取自定义元数据

您可以在卷中获取 Ceph 文件系统(CephFS)的自定义元数据、键值对，以及特定子卷组中的可选。

先决条件

- 一个正在运行的 Red Hat Ceph Storage 集群。
- 已创建 CephFS 卷、子卷组和子卷。
- 在 CephFS 子卷中创建的自定义元数据。

流程

- 获取 CephFS 子卷的元数据：

  语法

  ```
  ceph fs subvolume metadata get VOLUME_NAME SUBVOLUME_NAME KEY_NAME [--
  group_name SUBVOLUME_GROUP_NAME]
  ```

  示例

  ```
  [ceph: root@host01 /]# ceph fs subvolume metadata get cephfs sub0 test_meta --
  group_name subgroup0

  cluster
  ```

4.4.3. 列出文件系统子卷上的自定义元数据

您可以列出与卷中 Ceph 文件系统(CephFS)密钥关联的自定义元数据，也可以选择在特定子卷组中。

先决条件

- 一个正在运行的 Red Hat Ceph Storage 集群。
- 已创建 CephFS 卷、子卷组和子卷。
- 在 CephFS 子卷中创建的自定义元数据。

流程

- 列出 CephFS 子卷上的元数据：

  语法

  ```
  ceph fs subvolume metadata ls VOLUME_NAME SUBVOLUME_NAME [--group_name SUBVOLUME_GROUP_NAME]
  ```

  示例

  ```
  [ceph: root@host01 /]# ceph fs subvolume metadata ls cephfs sub0
  {
      "test_meta": "cluster"
  }
  ```

4.4.4. 从文件系统子卷中删除自定义元数据

您可以删除卷中 Ceph 文件系统(CephFS)的自定义元数据、键值对，以及可选的特定子卷组。

先决条件

- 一个正在运行的 Red Hat Ceph Storage 集群。
- 已创建 CephFS 卷、子卷组和子卷。
- 在 CephFS 子卷中创建的自定义元数据。

流程

1. 删除 CephFS 子卷上的自定义元数据：

   语法

   ```
   ceph fs subvolume metadata rm VOLUME_NAME SUBVOLUME_NAME KEY_NAME [--group_name SUBVOLUME_GROUP_NAME]
   ```

   示例

   ```
   [ceph: root@host01 /]# ceph fs subvolume metadata rm cephfs sub0 test_meta --group_name subgroup0
   ```

2. 列出元数据：

   示例

```
[ceph: root@host01 /]# ceph fs subvolume metadata ls cephfs sub0

{}
```

第 5 章 CEPH 文件系统管理

作为存储管理员，您可以执行常见的 Ceph 文件系统 (CephFS) 管理任务，例如：

- 实时监控 CephFS 指标，请参见 第 5.1 节 “使用 **cephfs-top** 工具”
- 要将目录映射到特定 MDS 等级，请参阅 第 5.5 节 “将目录树映射到元数据服务器守护进程等级”。
- 要从 MDS 等级中取消关联一个目录，请参阅 第 5.6 节 “与元数据服务器守护进程解除目录树的关联”。
- 添加新数据池，请参阅 第 5.7 节 “添加数据池”。
- 要使用配额，请参阅 第 7 章 *Ceph 文件系统配额*。
- 使用文件和目录布局，请参阅 第 8 章 *文件和目录布局*。
- 删除 Ceph 文件系统，请参阅 第 5.9 节 “删除 Ceph 文件系统”。
- 客户端功能，请参阅 第 5.11 节 “客户端特性”。
- 使用 **ceph mds fail** 命令，请参阅 第 5.10 节 “使用 **ceph mds fail** 命令”。
- 手动驱除 CephFS 客户端，请参阅 第 5.14 节 “手动驱除 Ceph 文件系统客户端”

先决条件

- 一个运行良好、健康的 Red Hat Ceph Storage 集群。
- 安装和配置 Ceph 元数据服务器守护进程 (**ceph-mds**)。
- 创建并挂载 Ceph 文件系统。

5.1. 使用 CEPHFS-TOP 工具

Ceph 文件系统(CephFS)提供了一个类似于 **top** 的实用程序，以实时显示 Ceph 文件系统上的指标。**cephfs-top** 实用程序是一个基于 **curses** 的 Python 脚本，它使用 Ceph Manager **stats** 模块来获取和显示客户端性能指标。

目前，**cephfs-top** 实用程序支持几乎 10k 客户端。

注意

目前，Red Hat Enterprise Linux 9.2 内核都不提供所有性能统计。在 Red Hat Enterprise Linux 9 及更高版本上支持 CephFS **-top**，并使用 Red Hat Enterprise Linux 中的其中一个标准终端。

重要

cephfs-top 实用程序的最小兼容 python 版本为 3.6.0。

先决条件

- 一个健康且运行 Red Hat Ceph Storage 集群。

- 部署 Ceph 文件系统.
- Ceph 客户端节点的根级别访问权限。
- 安装 **cephfs-top** 软件包。

流程

1. 如果还没有启用，启用 Red Hat Ceph Storage 7 工具存储库：

   Red Hat Enterprise Linux 9

   ```
   [root@client ~]# subscription-manager repos --enable=rhceph-7-tools-for-rhel-9-x86_64-rpms
   ```

2. 安装 **cephfs-top** 软件包：

   示例

   ```
   [root@client ~]# dnf install cephfs-top
   ```

3. 启用 Ceph Manager **stats** 插件：

   示例

   ```
   [root@client ~]# ceph mgr module enable stats
   ```

4. 创建 **client.fstop** Ceph 用户：

   示例

   ```
   [root@client ~]# ceph auth get-or-create client.fstop mon 'allow r' mds 'allow r' osd 'allow r' mgr 'allow r' > /etc/ceph/ceph.client.fstop.keyring
   ```

   注意

   （可选）使用 **--id** 参数指定与 **client.fstop** 以外的不同的 Ceph 用户。

5. 启动 **cephfs-top** 工具：

   示例

   ```
   [root@client ~]# cephfs-top
   cephfs-top - Wed Nov 30 15:26:05 2022

   All Filesystem Info
   Total Client(s): 4 - 3 FUSE, 1 kclient, 0 libcephfs
   COMMANDS: m - select a filesystem | s - sort menu | l - limit number of clients | r - reset to default | q - quit

     client_id mount_root chit(%) dlease(%) ofiles oicaps oinodes rtio(MB) raio(MB) rsp(MB/s) wtio(MB) waio(MB) wsp(MB/s) rlatavg(ms) rlatsd(ms) wlatavg(ms) wlatsd(ms) mlatavg(ms) mlatsd(ms) mount_point@host/addr
   ```

```
Filesystem: cephfs1 - 2 client(s)


 4500    /        100.0  100.0    0    751   0      0.0    0.0     0.0    578.13  0.03    0.0
N/A       N/A      N/A       N/A      N/A      N/A      N/A@example/192.168.1.4
 4501    /        100.0  0.0    0    1   0      0.0    0.0     0.0    0.0   0.0     0.0      0.0       0.0
0.0       0.0      0.41      0.0      /mnt/cephfs2@example/192.168.1.4

Filesystem: cephfs2 - 2 client(s)


 4512    /        100.0  0.0    0    1    0      0.0    0.0     0.0    0.0     0.0     0.0      0.0
0.0       0.0      0.0       0.4      0.0      /mnt/cephfs3@example/192.168.1.4
 4518    /        100.0  0.0    0    1    0      0.0    0.0     0.0    0.0     0.0     0.0      0.0
0.0       0.0      0.0       0.52     0.0      /mnt/cephfs4@example/192.168.1.4
```

5.1.1. cephfs-top 实用程序互动命令

选择特定的文件系统，并使用 **cephfs-top** 实用程序互动命令查看与该文件系统相关的指标。

m

描述

文件系统选择 ：显示要选择的文件系统菜单。

q

描述

退出 ：如果您位于主屏幕，且包含所有文件系统信息，请退出该实用程序。如果您没有在主屏幕，它会将您重新重定向到主屏幕。

s

描述

sort 字段选择 ：指定排序字段。'cap_hit' 是默认值。

l

描述

客户端限制 ：设置要显示的客户端数量的限制。

r

描述

reset ：重置排序字段，并将值限制为默认值。

可以使用箭头键、PgUp/PgDn、Home/End 和 mouse 滚动指标显示。

输入并退出文件系统选择菜单的示例

```
[root@client ~]# m

                    Filesystems
Press "q" to go back to home (all filesystem info) screen
```

```
                    cephfs01
                    cephfs02
[root@client ~]# q

cephfs-top - Thu Oct 20 07:29:35 2022
Total Client(s): 3 - 2 FUSE, 1 kclient, 0 libcephfs
```

5.1.2. cephfs-top 实用程序选项

您可以使用 **cephfs-top** utility 命令及各种选项。

示例

```
[root@client ~]# cephfs-top --selftest
selftest ok
```

--cluster *NAME_OF_THE_CLUSTER*

描述

使用此选项，您可以连接到非默认集群名称。默认名称为 **ceph**。

--id *USER*

描述

这是连接到 Ceph 集群的客户端，默认为 **fstop**。

--selftest

描述

通过此选项，您可以执行自我测试。此模式执行 **stats** 模块的健全检查。

--conffile *PATH_TO_THE_CONFIGURATION_FILE*

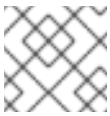
描述

通过此选项，您可以提供 Ceph 集群配置文件的路径。

-d/--delay *INTERVAL_IN_SECONDS*

描述

cephfs-top 实用程序默认每秒刷新统计。使用这个选项，您可以更改刷新间隔。



注意

间隔应大于或等于 1 秒。部分秒被接受。

--dump

描述

使用此选项，您可以在不创建策展显示的情况下将指标转储到 stdout。

--dumpfs *FILE_SYSTEM_NAME*

描述

使用此选项，您可以在不创建策展显示的情况下将给定文件系统的指标转储到 stdout。

5.2. 使用 MDS 自动缩放器模块

MDS Autoscaler 模块监控 Ceph 文件系统(CephFS)，以确保有足够的 MDS 守护进程可用。它的工作原理是调整 MDS 服务的 Orchestrator 后端的放置规格。

模块监控以下文件系统设置以告知放置计数：

- **max_mds** 文件系统设置
- **standby_count_wanted** 文件系统设置

Ceph 监控守护进程仍负责根据这些设置来提升或停止 MDS。**mds_autoscaler** 只调整由编配器生成的 MDS 数量。

先决条件

- 一个健康且运行 Red Hat Ceph Storage 集群。
- 部署 Ceph 文件系统.
- Ceph 监控节点的根级别访问权限.

流程

- 启用 MDS 自动缩放器模块：

  示例

  ```
  [ceph: root@host01 /]# ceph mgr module enable mds_autoscaler
  ```

5.3. 卸载挂载为内核客户端的 CEPH 文件系统

如何卸载挂载为内核客户端的 Ceph 文件系统。

先决条件

- 对执行挂载的节点的根级别访问权限。

流程

- 卸载挂载为内核客户端的 Ceph 文件系统：

  语法

  ```
  umount MOUNT_POINT
  ```

  示例

```
[root@client ~]# umount /mnt/cephfs
```

其它资源

- **umount(8)** 手册页

5.4. 卸载挂载为 FUSE 客户端的 CEPH 文件系统

卸载作为文件系统挂载到用户空间 (FUSE) 客户端中的 Ceph 文件系统。

先决条件

- 对 FUSE 客户端节点的根级别访问权限。

流程

- 卸载挂载在 FUSE 中的 Ceph 文件系统：

  语法

  ```
  fusermount -u MOUNT_POINT
  ```

  示例

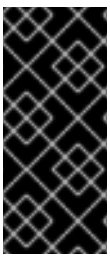
  ```
  [root@client ~]# fusermount -u /mnt/cephfs
  ```

其它资源

- **ceph-fuse(8)** 手册页

5.5. 将目录树映射到元数据服务器守护进程等级

要将目录及其子目录映射到特定的活动元数据服务器 (MDS) 排名，以使其元数据仅由持有该等级的 MDS 守护进程管理。这种方法允许您将应用程序负载或限制用户元数据请求的影响均匀分布到整个存储集群。



重要

内部均衡已经动态分散应用程序负载。因此，仅将目录树映射到某些精心选择的应用的排名上。

另外，当目录映射到等级时，平衡器无法分割它。因此，映射目录中的大量操作可能会过载等级和管理它的 MDS 守护进程。

先决条件

- 至少两个活跃的 MDS 守护进程。
- 用户访问 CephFS 客户端节点。
- 使用挂载的 Ceph 文件系统，验证 CephFS 客户端节点上已安装了 **attr** 软件包。

流程

1. 将 **p** 标志添加到 Ceph 用户的功能中：

   语法

   ```
   ceph fs authorize FILE_SYSTEM_NAME client.CLIENT_NAME /DIRECTORY CAPABILITY
   [/DIRECTORY CAPABILITY] ...
   ```

   示例

   ```
   [user@client ~]$ ceph fs authorize cephfs_a client.1 /temp rwp

   client.1
     key: AQBSdFhcGZFUDRAAcKhG9Cl2HPiDMMRv4DC43A==
     caps: [mds] allow r, allow rwp path=/temp
     caps: [mon] allow r
     caps: [osd] allow rw tag cephfs data=cephfs_a
   ```

2. 在目录中设置 **ceph.dir.pin** 扩展属性：

   语法

   ```
   setfattr -n ceph.dir.pin -v RANK DIRECTORY
   ```

   示例

   ```
   [user@client ~]$ setfattr -n ceph.dir.pin -v 2 /temp
   ```

   这个示例分配 **/temp** 目录及其所有子目录来等级 2。

其它资源

- 有关 **p** 标志的详情，请参见 *Red Hat Ceph Storage File System Guide* 中的 *Layout, quota, snapshot, and network restrictions* 部分。
- 如需更多详细信息，请参阅 *Red Hat Ceph Storage 文件系统指南* 中的*手动固定目录树*一节。
- 如需更多信息，请参阅 *Red Hat Ceph Storage 文件系统指南* 中的*配置多个活跃元数据服务器守护进程* 部分。

5.6. 与元数据服务器守护进程解除目录树的关联

将目录与特定的活动元数据服务器 (MDS) 解除关联。

先决条件

- 用户访问 Ceph 文件系统 (CephFS) 客户端节点.
- 确保在客户端节点上安装了 **attr** 软件包并带有一个挂载的 CephFS。

流程

- 在目录中将 **ceph.dir.pin** 扩展属性设置为 -1：

语法

```
setfattr -n ceph.dir.pin -v -1 DIRECTORY
```

示例

```
[user@client ~]$ setfattr -n ceph.dir.pin -v -1 /home/ceph-user
```

注意

任何单独映射的 **/home/ceph-user/** 子目录均不受影响。

其它资源

- 如需了解更多详细信息，请参阅 *Red Hat Ceph Storage 文件系统指南* 中的 *Mapping 目录树到元数据服务器守护进程排名*部分。

5.7. 添加数据池

Ceph 文件系统 (CephFS) 支持添加多个池来存储数据。这对以下情况非常有用：

- 在减少的冗余池中存储日志数据。
- 在 SSD 或 NVMe 池中存储用户主目录。
- 基本数据分隔。

在 Ceph 文件系统中使用另一个数据池之前，您必须添加它，如本节所述。

默认情况下，CephFS 使用创建期间指定的初始数据池来存储文件数据。要使用辅助数据池，还必须配置文件系统层次结构的一部分，以使用文件和目录布局将该池中或选择性地存储在该池的命名空间内存储文件数据。

先决条件

- Ceph 监控节点的根级别访问权限.

流程

1. 创建新数据池：

   语法

   ```
   ceph osd pool create POOL_NAME
   ```

   替换：

   - ***POOL_NAME***，池的名称。

   示例

```
[ceph: root@host01 /]# ceph osd pool create cephfs_data_ssd
pool 'cephfs_data_ssd' created
```

2. 在元数据服务器控制下添加新创建的池：

   语法

   ```
   ceph fs add_data_pool FS_NAME POOL_NAME
   ```

   替换：

   - ***FS_NAME***，文件系统的名称.
   - ***POOL_NAME***，池的名称。

   例如：

   ```
   [ceph: root@host01 /]# ceph fs add_data_pool cephfs cephfs_data_ssd
   added data pool 6 to fsmap
   ```

3. 验证池是否已成功添加：

   示例

   ```
   [ceph: root@host01 /]# ceph fs ls
   name: cephfs, metadata pool: cephfs_metadata, data pools: [cephfs_data cephfs_data_ssd]
   ```

4. 可选：从文件系统中删除数据池：

   语法

   ```
   ceph fs rm_data_pool FS_NAME POOL_NAME
   ```

   例如：

   ```
   [ceph: root@host01 /]# ceph fs rm_data_pool cephfs cephfs_data_ssd
   removed data pool 6 from fsmap
   ```

   a. 验证池是否已成功移除：

      示例

      ```
      [ceph: root@host01 /]# ceph fs ls
      name: cephfs, metadata pool: cephfs_metadata, data pools: [cephfs.cephfs.data]
      ```

5. 如果使用 **cephx** 身份验证，请确保客户端可以访问新的池。

其它资源

- 有关详细信息，请参阅 *Red Hat Ceph Storage 文件系统指南* 中的*文件和目录布局*部分。
- 如需更多详细信息，请参阅 *Red Hat Ceph Storage 文件系统指南* 中的*为 Ceph 文件系统创建客户端*部分。

5.8. 关闭 CEPH 文件系统集群

您可以通过将 **down** 标记设置为 **true** 来关闭 Ceph 文件系统(CephFS)集群。执行此操作将通过清空日志到元数据池并停止所有客户端 I/O，正常关闭元数据服务器 (MDS) 守护进程。

您还可以快速关闭 CephFS 集群来测试文件系统的删除，并使元数据服务器 (MDS) 守护进程停机，例如练习灾难恢复方案。这样做可设置 **jointable** 标志，以防止 MDS 备用守护进程激活文件系统。

先决条件

- Ceph 监控节点的根级别访问权限.

流程

1. 将 CephFS 集群标记为 down：

   语法

   ```
   ceph fs set FS_NAME down true
   ```

   示例

   ```
   [ceph: root@host01 /]# ceph fs set cephfs down true
   ```

   a. 使用 CephFS 集群备份：

      语法

      ```
      ceph fs set FS_NAME down false
      ```

      示例

      ```
      [ceph: root@host01 /]# ceph fs set cephfs down false
      ```

   或者

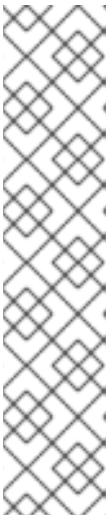
1. 快速关闭 CephFS 集群：

   语法

   ```
   ceph fs fail FS_NAME
   ```

   示例

   ```
   [ceph: root@host01 /]# ceph fs fail cephfs
   ```



注意

要将 CephFS 集群重新在线，将 **cephfs** 设置为 **joinable**:

语法

```
ceph fs set FS_NAME joinable true
```

示例

```
[ceph: root@host01 /]# ceph fs set cephfs joinable true

cephfs marked joinable; MDS may join as newly active.
```

5.9. 删除 CEPH 文件系统

您可以删除 Ceph 文件系统 (CephFS)。在进行此操作前，请考虑备份所有数据并验证所有客户端是否已在本地卸载该文件系统。

警告

此操作具有破坏性，将使 Ceph 文件系统上存储的数据永久无法访问。

先决条件

- 备份您的数据。
- Ceph 监控节点的根级别访问权限.

流程

1. 将存储集群标记为 down：

   语法

   ```
   ceph fs set FS_NAME down true
   ```

   替换

   - 将 *FS_NAME* 替换为您要删除的 Ceph 文件系统的名称。

   示例

   ```
   [ceph: root@host01 /]# ceph fs set cephfs down true

   cephfs marked down.
   ```

2. 显示 Ceph 文件系统的状态：

```
ceph fs status
```

示例

```
[ceph: root@host01 /]# ceph fs status

cephfs - 0 clients
======
+------------------+----------+-------+-------+
|       POOL       |   TYPE   | USED | AVAIL |
+-----------------+-----------+-------+-------+
|cephfs.cephfs.meta | metadata | 31.5M |  52.6G|
|cephfs.cephfs.data |   data   |    0  |  52.6G|
+-----------------+----------+-------+---------+
              STANDBY MDS
cephfs.ceph-host01
cephfs.ceph-host02
cephfs.ceph-host03
```

3. 删除 Ceph 文件系统：

语法

```
ceph fs rm FS_NAME --yes-i-really-mean-it
```

替换

- 将 *FS_NAME* 替换为您要删除的 Ceph 文件系统的名称。

示例

```
[ceph: root@host01 /]# ceph fs rm cephfs --yes-i-really-mean-it
```

4. 验证文件系统是否已成功删除：

示例

```
[ceph: root@host01 /]# ceph fs ls
```

5. 可选。删除与删除的文件系统关联的数据和元数据池。

其它资源

- 请参阅 *Red Hat Ceph Storage 策略指南* 中的 *删除池* 部分。

5.10. 使用 CEPH MDS FAIL 命令

使用 **ceph mds fail** 命令：

- 将 MDS 守护进程标记为失败。如果守护进程活跃，且有合适的备用守护进程可用，如果在禁用 **standby-replay** 配置后待机守护进程处于活动状态，则使用这个命令会强制切换到待机守护进程。通过禁用 **standby-replay** 守护进程，这会防止分配新的 **standby-replay** 守护进程。
- 重新启动正在运行的 MDS 守护进程。如果后台程序处于活动状态且有合适的备用后台程序可用，"失败"后台程序将变为备用后台程序。

先决条件

- 安装和配置 Ceph MDS 守护进程.

流程

- 守护进程失败：

  语法

  ```
  ceph mds fail MDS_NAME
  ```

  其中 *MDS_NAME* 是 **standby-replay** MDS 节点的名称。

  示例

  ```
  [ceph: root@host01 /]# ceph mds fail example01
  ```

  注意

  您可以从 **ceph fs status** 命令找到 Ceph MDS 名称。

其它资源

- 请参阅 *Red Hat Ceph Storage 文件系统指南* 中的*元数据服务器守护进程*部分的内容。
- 请参阅 *Red Hat Ceph Storage 文件系统指南* 中的*配置备用后台程序数量*部分。
- 请参阅 *Red Hat Ceph Storage 文件系统指南* 中的*元数据服务器排名*部分。

5.11. 客户端特性

有时，您可能希望设置 Ceph 文件系统(CephFS)功能，客户端必须支持它们才能使用 Ceph 文件系统。没有这些功能的客户端可能会破坏其他 CephFS 客户端，或以意外方式的行为。此外，您可能还需要新的功能来防止较早的功能，并且可能会需要 buggy 客户端连接到 Ceph 文件系统。

重要

CephFS 客户端缺少新添加的功能会自动被驱除。

您可以使用 **fs features ls** 命令列出所有 CephFS 功能。您可以使用 **fs required_client_features** 命令添加或删除要求。

语法

```
fs required_client_features FILE_SYSTEM_NAME add FEATURE_NAME
fs required_client_features FILE_SYSTEM_NAME rm FEATURE_NAME
```

功能描述

reply_encoding

描述

如果客户端支持此功能，Ceph 元数据服务器(MDS)以可扩展格式对回复请求进行编码。

reclaim_client

描述

Ceph MDS 允许新客户端回收另一个客户端的状态。NFS Ganesha 使用此功能。

lazy_caps_wanted

描述

当过时的客户端恢复时，如果客户端支持此功能，Ceph MDS 只需要重新发布明确需要的功能。

multi_reconnect

描述

在 Ceph MDS 故障转移事件后，客户端向 MDS 发送重新连接消息，以重新建立缓存状态。客户端可以将大量重新连接消息分成多个消息。

deleg_ino

描述

如果客户端支持此功能，Ceph MDS 会将索引节点编号委派给客户端。委派索引节点编号是客户端进行同步文件的先决条件。

metric_collect

描述

CephFS 客户端可以将性能指标发送到 Ceph MDS。

alternate_name

描述

CephFS 客户端可以设置和理解目录条目的备用名称。此功能允许加密文件名。

5.12. CEPH 文件系统客户端驱除

当 Ceph 文件系统 (CephFS) 客户端不响应或行为不当时，可能需要强制终止或驱逐它访问 CephFS。驱除 CephFS 客户端会阻止它进一步与元数据服务器 (MDS) 守护进程和 Ceph OSD 守护进程通信。如果 CephFS 客户端在驱除时将 I/O 缓冲到 CephFS，则任何未清空的数据都将丢失。CephFS 客户端驱除过程适用于所有客户端类型：FUSE 挂载、内核挂载、NFS 网关，以及使用 **libcephfs** API 库的任何进程。

如果 CephFS 客户端无法及时与 MDS 守护进程通信或手动通信，您可以自动驱除 CephFS 客户端。

自动客户端驱除

这些场景导致自动 CephFS 客户端驱除：

- 如果 CephFS 客户端在默认的 300 秒内未与活动 MDS 守护进程通信，或者与 **session_autoclose** 选项所设置的 MDS 守护进程通信。
- 如果设置了 **mds_cap_revoke_eviction_timeout** 选项，并且 CephFS 客户端没有在设定的秒数内响应最大撤销的消息。默认情况下禁用 **mds_cap_revoke_eviction_timeout** 选项。
- 在 MDS 启动或故障转移期间，MDS 守护进程经过一个重新连接阶段，等待所有 CephFS 客户端连接到新的 MDS 守护进程。如果有任何 CephFS 客户端无法在默认时间窗内重新连接 45 秒，或由 **mds_reconnect_timeout** 选项设置。

其它资源

- 如需了解更多详细信息，请参阅 *Red Hat Ceph Storage File System Guide* 中的 *Manually evicting a Ceph File System client* 部分。

5.13. BLOCKLIST CEPH 文件系统客户端

Ceph 文件系统(CephFS)客户端阻止列表默认启用。当您向单个元数据服务器 (MDS) 守护进程发送驱除命令时，它会将黑名单传播到其他 MDS 守护进程。这是为了防止 CephFS 客户端访问任何数据对象，因此需要更新其他 CephFS 客户端，并且使用最新的 Ceph OSD map（包含黑名单的客户端条目）更新 MDS 守护进程。

在更新 Ceph OSD map 时，使用内部"osdmap epoch 障碍"机制。障碍的目的是验证接受功能的 CephFS 客户端具有足够最新的 Ceph OSD map，然后分配任何允许访问同一 RADOS 对象的功能，以免对已取消的操作（如 ENOSPC 或来自驱除的客户端列入黑名单）争用。

如果您由于节点缓慢或网络不可靠而频繁遇到 CephFS 客户端驱除，并且您无法修复底层问题，则您可以要求 MDS 不太严格。可以通过丢弃 MDS 会话来响应缓慢的 CephFS 客户端，但允许 CephFS 客户端重新打开会话并继续与 Ceph OSD 交互。通过将 **mds_session_blocklist_on_timeout** 和 **mds_session_blocklist_on_evict** 选项设置为 **false** 可启用这个模式。

注意

禁用块列表时，被驱除的 CephFS 客户端仅对发送该命令的 MDS 守护进程产生影响。在具有多个活跃 MDS 守护进程的系统上，您需要向每个活跃的守护进程发送驱除命令。

5.14. 手动驱除 CEPH 文件系统客户端

如果客户端的行为不当，且您无法访问客户端节点，或者客户端出现故障，并且您不希望等待客户端会话超时，您可能希望手动驱除 Ceph 文件系统 (CephFS) 客户端。

先决条件

- Ceph 监控节点的根级别访问权限.

流程

1. 查看客户端列表：

   语法

   ```
   ceph tell DAEMON_NAME client ls
   ```

   示例

```
[ceph: root@host01 /]# ceph tell mds.0 client ls
[
    {
        "id": 4305,
        "num_leases": 0,
        "num_caps": 3,
        "state": "open",
        "replay_requests": 0,
        "completed_requests": 0,
        "reconnecting": false,
        "inst": "client.4305 172.21.9.34:0/422650892",
        "client_metadata": {
            "ceph_sha1": "79f0367338897c8c6d9805eb8c9ad24af0dcd9c7",
            "ceph_version": "ceph version 16.2.8-65.el8cp
(79f0367338897c8c6d9805eb8c9ad24af0dcd9c7)",
            "entity_id": "0",
            "hostname": "senta04",
            "mount_point": "/tmp/tmpcMpF1b/mnt.0",
            "pid": "29377",
            "root": "/"
        }
    }
]
```

2. 驱除指定的 CephFS 客户端：

语法

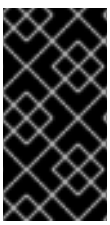
```
ceph tell DAEMON_NAME client evict id=ID_NUMBER
```

示例

```
[ceph: root@host01 /]# ceph tell mds.0 client evict id=4305
```

5.15. 从块列表中删除 CEPH 文件系统客户端

在某些情况下，允许之前列入黑名单的 Ceph 文件系统 (CephFS) 客户端重新连接到存储集群非常有用。



重要

从黑名单中删除 CephFS 客户端会使数据完整性面临风险，也无法保证完全正常运行的 CephFS 客户端。在驱除后重新获取完全健康的 CephFS 客户端的最佳方法是卸载 CephFS 客户端并执行全新的挂载。如果其他 CephFS 客户端正在访问阻止的 CephFS 客户端被缓冲到的文件，则可能会导致数据崩溃。

先决条件

- Ceph 监控节点的根级别访问权限.

流程

1. 查看黑名单：

示例

```
[ceph: root@host01 /]# ceph osd blocklist ls

listed 1 entries
127.0.0.1:0/3710147553 2022-05-09 11:32:24.716146
```

2. 从黑名单中删除 CephFS 客户端：

语法

```
ceph osd blocklist rm CLIENT_NAME_OR_IP_ADDR
```

示例

```
[ceph: root@host01 /]# ceph osd blocklist rm 127.0.0.1:0/3710147553

un-blocklisting 127.0.0.1:0/3710147553
```

3. 另外，您还可以在从 blocklist 中删除时，自动重新连接基于内核的 CephFS 客户端。在基于内核的 CephFS 客户端中，将以下选项设置为在手动挂载时**清除**，或使用 **/etc/fstab** 文件中的条目自动挂载：

```
recover_session=clean
```

4. （可选）您可以在将基于 FUSE 的 CephFS 客户端从 blocklist 中删除时自动重新连接。在 FUSE 客户端上，在进行手动挂载时，将以下选项设置为 **true**，或者在 **/etc/fstab** 文件中使用条目自动挂载：

```
client_reconnect_stale=true
```

其它资源

- 如需了解更多详细信息，请参阅 *Red Hat Ceph Storage 文件系统指南* 中的*将 Ceph 文件系统挂载为 FUSE 客户端*部分。

第 6 章 NFS 集群和导出管理

作为存储管理员，您可以创建一个 NFS 集群，对其进行自定义，并通过 NFS 协议导出 Ceph 文件系统命名空间。

先决条件

- 一个运行良好、健康的 Red Hat Ceph Storage 集群。
- 安装和配置 Ceph 元数据服务器守护进程 (**ceph-mds**)。
- 创建并挂载 Ceph 文件系统。

6.1. 创建一个 NFS 集群

使用 **nfs cluster create** 命令创建 NFS 集群。这会为所有 NFS Ganesha 守护进程、基于集群名称和通用 NFS Ganesha 配置 RADOS 对象的新用户创建通用恢复池。

先决条件

- 一个运行良好、健康的 Red Hat Ceph Storage 集群。
- 现有的 Ceph 文件系统.
- Ceph 监控器的 root 级别访问。
- 在 Ceph 管理器主机上安装 **nfs-ganesha**、**nfs-ganesha-ceph**、**nfs-ganesha-rados-grace** 和 **nfs-ganesha-rados-urls** 软件包。
- 对客户端的 root 级别访问。

流程

1. 登录到 Cephadm shell：

   示例

   ```
   [root@mds ~]# cephadm shell
   ```

2. 启用 Ceph Manager NFS 模块：

   示例

   ```
   [ceph: root@host01 /]# ceph mgr module enable nfs
   ```

3. 创建 NFS Ganesha 集群：

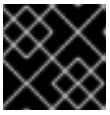
   语法

   ```
   ceph nfs cluster create CLUSTER_NAME [PLACEMENT] [--ingress] [--virtual_ip
   IP_ADDRESS] [--ingress-mode {default|keepalive-only|haproxy-standard|haproxy-protocol}]
   [--port PORT]
   ```

示例

```
[ceph: root@host01 /]# ceph nfs cluster create nfs-cephfs "host01 host02"
NFS Cluster Created Successfully
```

在本例中，NFS Ganesha 集群名称为 **nfs-cephfs**，守护进程容器则部署到 **host01**，和 **host02**。



重要

红帽只支持每个主机运行一个 NFS Ganesha 守护进程。

4. 验证 NFS Ganesha 集群信息：

语法

```
ceph nfs cluster info [CLUSTER_NAME]
```

示例

```
[ceph: root@host01 /]# ceph nfs cluster info nfs-cephfs
{
   "nfs-cephfs": [
      {
         "hostname": "host01",
         "ip": "10.74.179.124",
         "port": 2049
      },
      {
         "hostname": "host02",
         "ip": "10.74.180.160",
         "port": 2049
      }
   ]
}
```

注意

指定 *CLUSTER_NAME* 是可选的。

6.2. 自定义 NFS 配置

使用配置文件自定义 NFS 集群。因此，NFS 集群使用指定的配置，并优先于默认配置块。

先决条件

- 一个运行良好、健康的 Red Hat Ceph Storage 集群。
- 对 Ceph 元数据服务器(MDS)节点的根级访问。
- 使用 **ceph nfs cluster create** 命令创建的 NFS 集群。

流程

1. 创建配置文件：

   示例

   ```
   [ceph: root@host01 /]# touch nfs-cephfs.conf
   ```

2. 使用以下块在配置文件中启用日志记录：

   示例

   ```
   [ceph: root@host01 /]# vi nfs-cephfs.conf

   LOG {
    COMPONENTS {
       ALL = FULL_DEBUG;
    }
   }
   ```

3. 设置新配置：

   语法

   ```
   ceph nfs cluster config set CLUSTER_NAME -i PATH_TO_CONFIG_FILE
   ```

   示例

   ```
   [ceph: root@host01 /]# ceph nfs cluster config set nfs-cephfs -i nfs-cephfs.conf

   NFS-Ganesha Config Set Successfully
   ```

4. 查看自定义的 NFS Ganesha 配置：

   语法

   ```
   ceph nfs cluster config get CLUSTER_NAME
   ```

   示例

   ```
   [ceph: root@host01 /]#  ceph nfs cluster config get nfs-cephfs

   LOG {
      COMPONENTS {
         ALL = FULL_DEBUG;
      }
   }
   ```

   这将提供用户定义的配置的输出（若有）。

5. 可选：如果要删除用户定义的配置，请运行以下命令：

   语法

```
ceph nfs cluster config reset CLUSTER_NAME
```

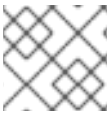
示例

```
[ceph: root@host01 /]# ceph nfs cluster config reset nfs-cephfs

NFS-Ganesha Config Reset Successfully
```

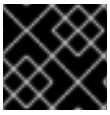
6.3. 通过 NFS 协议导出 CEPH 文件系统命名空间（有限的可用性）

Ceph 文件系统(CephFS)命名空间可通过 NFS 协议使用 NFS Ganesha 文件服务器导出。要导出 CephFS 命名空间，您必须首先有一个正在运行的 NFS Ganesha 集群。



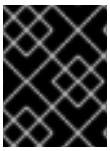
注意

该技术是有限可用性。如需更多信息，请参阅 *已弃用的功能* 章节。



重要

红帽只支持 NFS 版本 4.0 或更高版本。



重要

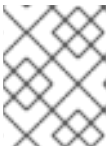
NFS 客户端无法通过其原生 NFS 挂载创建 CephFS 快照。它们必须使用服务器端操作器工具来满足其快照需求。

先决条件

- 一个运行良好、健康的 Red Hat Ceph Storage 集群。
- 使用 **ceph nfs cluster create** 命令创建的 NFS 集群。

流程

1. 创建 CephFS 导出：



注意

在创建 NFS 导出时不要使用 **cmount_path** 选项。这是因为，如果在使用 '/' 以外的任何其他值的 **cmount_path**，则之前定义的 NFS 导出变得不可访问。

语法

```
ceph nfs export create cephfs CLUSTER_NAME BINDING FILE_SYSTEM_NAME [--readonly] [--path=PATH_WITHIN_CEPHFS]
```

示例

```
[ceph: root@host01 /]# ceph nfs export create cephfs nfs-cephfs /ceph cephfs01 --path=/

{
```

```
  "bind": "/ceph",
  "cluster": "nfs-cephfs",
  "fs": "cephfs01",
  "mode": "RW",
  "path": "/"
}
```

在本例中，*BINDING* (/**ceph**)是伪根路径，该路径必须是唯一且绝对路径。

注意

--readonly 选项导出具有只读权限的路径，默认是读取和写入权限。

注意

PATH_WITHIN_CEPHFS 可以是一个子卷。您可以使用以下命令获取绝对路径：

语法

```
ceph fs subvolume getpath VOLUME_NAME SUBVOLUME_NAME [--
group_name SUBVOLUME_GROUP_NAME]
```

示例

```
[ceph: root@host01 /]# ceph fs subvolume getpath cephfs sub0
```

2. 根据伪根名称查看导出块：

语法

```
ceph nfs export get CLUSTER_NAME BINDING
```

示例

```
[ceph: root@host01 /]# ceph nfs export get nfs-cephfs /ceph
{
  "export_id": 1,
  "path": "/",
  "cluster_id": "nfs-cephfs",
  "pseudo": "/ceph",
  "access_type": "RW",
  "squash": "no_root_squash",
  "security_label": true,
  "protocols": [
    4
  ],
  "transports": [
    "TCP"
  ],
  "fsal": {
    "name": "CEPH",
    "user_id": "cephnfs11",
    "fs_name": "cephfs",
```

```
    "sec_label_xattr": ""
  },
  "clients": []
}
```

3. 列出 NFS 导出：

语法

```
ceph nfs export ls CLUSTER_NAME [--detailed]
```

示例

```
[ceph: root@host01 /]# ceph nfs export ls nfs-cephfs
[
  "/ceph/"
]
[ceph: root@host01 /]# ceph nfs export ls nfs-cephfs --detailed
[
  {
    "export_id": 100,
    "path": "/",
    "cluster_id": "nfs-cephfs",
    "pseudo": "/ceph/",
    "access_type": "RW",
    "squash": "no_root_squash",
    "security_label": true,
    "protocols": [
      4
    ],
    "transports": [
      "TCP"
    ],
    "fsal": {
      "name": "CEPH",
      "user_id": "nfstest01",
      "fs_name": "cephfs",
      "sec_label_xattr": ""
    },
    "clients": []
  }
]
```

4. 获取 NFS 导出的信息：

语法

```
ceph nfs export info CLUSTER_NAME [PSEUDO_PATH]
```

示例

```
[ceph: root@host01 /]# ceph nfs export info nfs-cephfs /ceph

{
```

```
 "export_id": 1,
 "path": "/",
 "cluster_id": "nfs-cephfs",
 "pseudo": "/ceph",
 "access_type": "RW",
 "squash": "none",
 "security_label": true,
 "protocols": [
  4
 ],
 "transports": [
  "TCP"
 ],
 "fsal": {
  "name": "CEPH",
  "user_id": "nfs.nfs-cephfs.1",
  "fs_name": "cephfs"
 },
 "clients": []
}
```

5. 在客户端主机上挂载导出的 Ceph 文件系统：

语法

```
mount -t nfs -o port=GANESHA_PORT HOST_NAME:BINDING LOCAL_MOUNT_POINT
```

示例

```
[root@client01 ~]# mount -t nfs -o port=2049 host01:/ceph/ /mnt/nfs-cephfs
```

a. 要在启动时自动挂载，请通过添加新行来打开并编辑 **/etc/fstab** 文件：

语法

```
HOST_NAME:BINDING  LOCAL_MOUNT_POINT  nfs4
defaults,seclabel,vers=4.2,proto=tcp,port=2049  0  0
```

示例

```
host01:/ceph/  /mnt/nfs-cephfs  nfs4  defaults,seclabel,vers=4.2,proto=tcp,port=2049  0
0
```

6. 在客户端主机上挂载使用 **入口** 服务创建的导出的 NFS Ceph 文件系统：

语法

```
mount -t nfs VIRTUAL_IP_ADDRESS:BINDING LOCAL_MOUNT_POINT
```

- 将 *VIRTUAL_IP_ADDRESS* 替换为用于创建 NFS 集群的 **--ingress --virtual-ip** IP 地址。
- 使用伪根路径替换 *BINDING*。

- 将 *LOCAL_MOUNT_POINT* 替换为挂载导出的挂载点。

  示例

  ```
  [root@client01 ~]# mount -t nfs 10.10.128.75:/nfs-cephfs /mnt
  ```

  本例将导出 **nfs-cephfs** 挂载到使用 **--ingress --virtual-ip 10.10.128.75** 创建的 NFS 集群上，该集群位于挂载点 **/mnt** 上。

6.4. 修改 CEPH 文件系统导出

您可以使用配置文件修改导出中的以下参数：

- **access_type** - 这可以是 **RW**、**RO** 或 **NONE**。
- **squash** - 这可以是 **No_Root_Squash**、**None** 或 **Root_Squash**。
- **security_label** - 可以是 **true** 或 **false**。

先决条件

- 一个运行良好、健康的 Red Hat Ceph Storage 集群。
- NFS 导出已创建。

流程

1. 根据伪根名称查看导出块：

   语法

   ```
   ceph nfs export get CLUSTER_NAME BINDING
   ```

   示例

   ```
   [ceph: root@host01 /]# ceph nfs export get nfs-cephfs /ceph
   {
     "export_id": 1,
     "path": "/",
     "cluster_id": "nfs-cephfs",
     "pseudo": "/ceph",
     "access_type": "RO",
     "squash": "none",
     "security_label": true,
     "protocols": [
       4
     ],
     "transports": [
       "TCP"
     ],
     "fsal": {
       "name": "CEPH",
       "user_id": "cephnfs11",
       "fs_name": "cephfs",
   ```

```
    "sec_label_xattr": ""
  },
  "clients": []
}
```

2. 导出配置文件：

示例

```
[ceph: root@host01 /]# ceph nfs export get nfs-cephfs /ceph > export.conf
```

3. 编辑导出信息：

语法

```
{
  "export_id": EXPORT_ID,
  "path": "/",
  "cluster_id": "CLUSTER_NAME",
  "pseudo": "CLUSTER_PSEUDO_PATH",
  "access_type": "RW/RO",
  "squash": "SQUASH",
  "security_label": SECURITY_LABEL,
  "protocols": [
    PROTOCOL_ID_
  ],
  "transports": [
    "TCP"
  ],
  "fsal": {
    "name": "NAME",
    "user_id": "USER_ID",
    "fs_name": "FILE_SYSTEM_NAME",
    "sec_label_xattr": ""
  },
  "clients": []
}
```

示例

```
[ceph: root@host01 /]# vi export.conf

{
  "export_id": 1,
  "path": "/",
  "cluster_id": "nfs-cephfs",
  "pseudo": "/ceph",
  "access_type": "RW",
  "squash": "none",
  "security_label": true,
  "protocols": [
    4
  ],
  "transports": [
```

```
    "TCP"
  ],
  "fsal": {
    "name": "CEPH",
    "user_id": "cephnfs11",
    "fs_name": "cephfs",
    "sec_label_xattr": ""
  },
  "clients": []
}
```

在上例中，**access_type** 从 **RO** 修改为 **RW**。

4. 应用规格：

语法

```
ceph nfs export apply CLUSTER_NAME PATH_TO_EXPORT_FILE
```

示例

```
[ceph: root@host01 /]# ceph nfs export apply nfs-cephfs -i export.conf
Added export /ceph
```

5. 获取更新的导出信息：

语法

```
ceph nfs export get CLUSTER_NAME BINDING
```

示例

```
[ceph: root@host01 /]# ceph nfs export get nfs-cephfs /ceph
{
  "export_id": 1,
  "path": "/",
  "cluster_id": "nfs-cephfs",
  "pseudo": "/ceph",
  "access_type": "RW",
  "squash": "none",
  "security_label": true,
  "protocols": [
    4
  ],
  "transports": [
    "TCP"
  ],
  "fsal": {
    "name": "CEPH",
    "user_id": "cephnfs11",
    "fs_name": "cephfs",
    "sec_label_xattr": ""
```

```
  },
  "clients": []
}
```

6.5. 创建自定义 CEPH 文件系统导出

您可以自定义 Ceph 文件系统(CepFS)导出并应用配置。

先决条件

- 一个运行良好、健康的 Red Hat Ceph Storage 集群。
- 使用 **ceph nfs cluster create** 命令创建的 NFS 集群。
- 已创建 CephFS。

流程

1. 创建自定义文件：

   示例

   ```
   [ceph: root@host01 /]# touch export_new.conf
   ```

2. 使用自定义文件创建导出：

   语法

   ```
   EXPORT {
     Export_Id = EXPORT_ID;
     Transports = TCP/UDP;
     Path = PATH;
     Pseudo = PSEUDO_PATH;
     Protocols = NFS_PROTOCOLS;
     Access_Type = ACCESS_TYPE;
     Attr_Expiration_Time = EXPIRATION_TIME;
     Squash = SQUASH;
     FSAL {
       Name = NAME;
       Filesystem = "CEPH_FILE_SYSTEM_NAME";
       User_Id = "USER_ID";
     }
   }
   ```

   示例

   ```
   [ceph: root@host01 /]# cat export_new.conf
   EXPORT {
     Export_Id = 2;
     Transports = TCP;
     Path = /;
     Pseudo = /ceph1/;
     Protocols = 4;
   ```

```
  Access_Type = RW;
  Attr_Expiration_Time = 0;
  Squash = None;
  FSAL {
    Name = CEPH;
    Filesystem = "cephfs";
    User_Id = "nfs.nfs-cephfs.2";
  }
}
```

3. 应用规格：

语法

```
ceph nfs export apply CLUSTER_NAME -i PATH_TO_EXPORT_FILE
```

示例

```
[ceph: root@host01 /]# ceph nfs export apply nfs-cephfs -i new_export.conf
Added export /ceph1
```

4. 获取更新的导出信息：

语法

```
ceph nfs export get CLUSTER_NAME BINDING
```

示例

```
[ceph: root@host01 /]# ceph nfs export get nfs-cephfs /ceph1
{
  "export_id": 1,
  "path": "/",
  "cluster_id": "nfs-cephfs",
  "pseudo": "/ceph1",
  "access_type": "RW",
  "squash": "None",
  "security_label": true,
  "protocols": [
    4
  ],
  "transports": [
    "TCP"
  ],
  "fsal": {
    "name": "CEPH",
    "user_id": "nfs.nfs-cephfs.2",
    "fs_name": "cephfs",
    "sec_label_xattr": ""
  },
  "clients": []
}
```

6.6. 删除 CEPH 文件系统导出

您可以使用 **ceph export rm** 命令删除 Ceph 文件系统(CephFS) NFS 导出。

先决条件

- 一个运行良好、健康的 Red Hat Ceph Storage 集群。
- 已创建 CephFS。

流程

- 删除 CephFS 导出：

  语法

  ```
  ceph nfs export rm CLUSTER_NAME BINDING
  ```

  示例

  ```
  [ceph: root@host01 /]# ceph nfs export rm nfs-cephfs /ceph
  ```

6.7. 删除 NFS 集群

使用 **nfs cluster rm** 命令删除 NFS 集群。这会删除部署的集群。删除 NFS 守护进程和入口服务是异步的。使用 **ceph orch ls** 命令检查移除的状态。

先决条件

- 一个运行良好、健康的 Red Hat Ceph Storage 集群。
- 对 Ceph 元数据服务器(MDS)节点的根级访问。
- 使用 **ceph nfs cluster create** 命令部署的 NFS 守护进程。

流程

1. 登录到 Cephadm shell：

   示例

   ```
   [root@mds ~]# cephadm shell
   ```

2. 删除 NFS Ganesha 集群：

   语法

   ```
   ceph nfs cluster rm CLUSTER_NAME
   ```

   示例

   ```
   [ceph: root@host01 /]# ceph nfs cluster rm nfs-cephfs
   ```

第 7 章 CEPH 文件系统配额

作为存储管理员，您可以查看、设置和删除文件系统中任何目录的配额。您可以在 目录中对字节数或文件数施加配额限制。

先决条件

- 一个运行良好、健康的 Red Hat Ceph Storage 集群。
- 部署 Ceph 文件系统.
- 确保已安装了 **attr** 软件包。

7.1. CEPH 文件系统配额

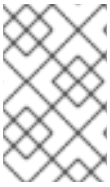
Ceph 文件系统 (CephFS) 配额允许您限制存储在目录结构中的字节数或文件数。Ceph 文件系统配额通过 FUSE 客户端或使用内核客户端（版本 4.17 或更新版本）完全支持。

限制

- CephFS 配额依赖于客户端挂载文件系统，从而在达到配置的限制时停止写入数据。但是，仅使用配额并不能阻止对立的、不受信任的客户端填充文件系统。
- 当向文件系统写入数据的进程达到配置的限制后，在数据量达到配额限制和进程停止写入数据之间相隔短暂的时间。时间段通常以秒为单位测量。但是，在该时间内，进程会继续写入数据。进程写入的额外数据量取决于进程停止前经过的时间量。
- 在使用基于路径的访问限制时，请确保在限制客户端的目录或嵌套在它下的目录上配置配额。如果客户端的访问权限受限于基于 MDS 能力的特定路径，并且配额是在客户端无法访问的上级目录中配置的，则客户端不会强制实施配额。例如，如果客户端无法访问 **/home/** 目录，且在 **/home/** 上配置了配额，客户端就无法强制设置目录 **/home/user/** 目录的配额。
- 已删除或更改的快照文件数据不会计算配额数。
- 在使用 **setxattr** 时，不支持带有 NFS 客户端的配额，且不支持 NFS 中文件级别配额。要在 NFS 共享中使用配额，您可以使用子卷导出配额并设置 **--size** 选项。

7.2. 查看配额

使用 **getfattr** 命令和 **ceph.quota** 扩展属性来查看目录的配额设置。



注意

如果属性出现在目录索引节点中，则该目录具有配置的配额。如果索引节点中未显示这些属性，则该目录没有设置配额，尽管其父目录可能已配置了配额。如果扩展属性的值为 **0**，则不设置配额。

先决条件

- 对 Ceph 客户端节点的根级别访问权限。
- 已安装 **attr** 软件包。

流程

1. 查看 CephFS 配额：

   a. 使用字节限制配额：

      语法

      ```
      getfattr -n ceph.quota.max_bytes DIRECTORY
      ```

      示例

      ```
      [root@client ~]# getfattr -n ceph.quota.max_bytes /mnt/cephfs/
      getfattr: Removing leading '/' from absolute path names
      # file: mnt/cephfs/
      ceph.quota.max_bytes="100000000"
      ```

      在这个示例中，**100000000** 等于 100 MB。

   b. 使用 file-limit 配额：

      语法

      ```
      getfattr -n ceph.quota.max_files DIRECTORY
      ```

      示例

      ```
      [root@client ~]# getfattr -n ceph.quota.max_files /mnt/cephfs/
      getfattr: Removing leading '/' from absolute path names
      # file: mnt/cephfs/
      ceph.quota.max_files="10000"
      ```

      在本例中，**10000** 等于 10,000 个文件。

其它资源

- 如需更多信息，请参阅 **getfattr(1)** 手册页。

7.3. 设置配额

本节介绍如何使用 **setfattr** 命令和 **ceph.quota** 扩展属性为目录设置配额。

先决条件

- 对 Ceph 客户端节点的根级别访问权限。
- 已安装 **attr** 软件包。

流程

- 使用字节限制配额为 direcotry 设置配额：

注意

以下值支持 byte-limit 配额：K、K、Ki、M、Mi、G、Gi、T 和 Ti。

语法

```
setfattr -n ceph.quota.max_bytes -v LIMIT_VALUE DIRECTORY
```

示例

```
[root@client ~]# setfattr -n ceph.quota.max_bytes -v 2T /cephfs/
```

- 使用 file-limit 配额为目录设置配额：

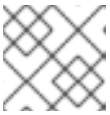
语法

```
setfattr -n ceph.quota.max_files -v LIMIT_VALUE DIRECTORY
```

示例

```
[root@client ~]# setfattr -n ceph.quota.max_files -v 10000 /cephfs/
```

在本例中，**10000** 等于 10,000 个文件。



注意

文件 *LIMIT_VALUE* 仅支持数字值。

其它资源

- 如需更多信息，请参阅 **setfattr(1)** 手册页。

7.4. 删除配额

本节介绍如何使用 **setfattr** 命令和 **ceph.quota** 扩展属性从目录中移除配额。

先决条件

- 对 Ceph 客户端节点的根级别访问权限。
- 确保已安装了 **attr** 软件包。

流程

1. 移除 CephFS 配额：
   a. 使用字节限制配额：

语法

```
setfattr -n ceph.quota.max_bytes -v 0 DIRECTORY
```

示例

```
[root@client ~]# setfattr -n ceph.quota.max_bytes -v 0 /mnt/cephfs/
```

b. 使用 file-limit 配额：

语法

```
setfattr -n ceph.quota.max_files -v 0 DIRECTORY
```

示例

```
[root@client ~]# setfattr -n ceph.quota.max_files -v 0 /mnt/cephfs/
```

其它资源

- 如需更多信息，请参阅 **setfattr(1)** 手册页。

其它资源

- 请参阅 *Red Hat Ceph Storage File System Guide* 中的 *部署 Ceph 文件系统*一节。
- 如需更多信息，请参阅 **getfattr(1)** 手册页。
- 如需更多信息，请参阅 **setfattr(1)** 手册页。

第 8 章 文件和目录布局

作为存储管理员，您可以控制文件或目录数据如何映射到对象。

本节描述了如何：

- 了解文件和目录布局
- 设置文件和目录布局
- 查看文件和目录布局字段
- 查看单个布局字段
- 删除目录布局

先决条件

- 一个运行良好、健康的 Red Hat Ceph Storage 集群。
- 部署 Ceph 文件系统.
- 安装 **attr** 软件包。

8.1. 文件和目录布局概述

本节介绍 Ceph 文件系统上下文中的文件和目录布局。

文件或目录的布局控制其内容如何映射到 Ceph RADOS 对象。目录布局主要用于为该目录中新文件设置继承布局。

要查看和设置文件或目录布局，请使用虚拟扩展属性或扩展文件属性 (**xattrs**)。布局属性的名称取决于文件是常规文件还是目录：

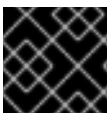
- 常规文件布局属性称为 **ceph.file.layout**。
- 目录布局属性称为 **ceph.dir.layout**。

布局继承

在创建文件时，文件会继承其父目录的布局。但是，随后对父目录布局的更改不会影响子级。如果目录没有设置任何布局，文件将使用目录结构中的布局从最接近的目录继承布局。

8.2. 设置文件和目录布局字段

使用 **setfattr** 命令设置文件或目录上的布局字段。



重要

当您修改文件的布局字段时，该文件必须为空，否则会出现错误。

先决条件

- 节点的根级别访问权限。

流程

- 修改文件或目录中的布局字段：

  语法

  ```
  setfattr -n ceph.TYPE.layout.FIELD -v VALUE PATH
  ```

  替换：

  - *TYPE*，**file** 或 **dir**。
  - *FIELD*，字段的名称。
  - *VALUE*，字段的新值 。
  - 带有到文件或目录路径的 *PATH*。

  示例

  ```
  [root@mon ~]# setfattr -n ceph.file.layout.stripe_unit -v 1048576 test
  ```

其它资源

- 如需了解更多详细信息，请参见 *Red Hat Ceph Storage* 的*文件和目录布局概述*部分中的表格。
- 请参阅 **setfattr(1)** 手册页。

8.3. 查看文件和目录布局字段

要使用 **getfattr** 命令查看文件或目录的布局字段：

先决条件

- 一个正在运行的 Red Hat Ceph Storage 集群。
- 对存储集群中所有节点的根级别访问权限。

流程

- 以单一字符串形式查看文件或目录中的布局字段：

  语法

  ```
  getfattr -n ceph.TYPE.layout PATH
  ```

  替换

  - 带有到文件或目录路径的 *PATH*。
  - *TYPE*，**file** 或 **dir**。

  示例

```
[root@mon ~]# getfattr -n ceph.dir.layout /home/test
ceph.dir.layout="stripe_unit=4194304 stripe_count=2 object_size=4194304
pool=cephfs_data"
```

注意

在设置目录之前，该目录不具有明确的布局。因此，尝试在没有首先设置的情况下查看布局会失败，因为没有更改。

其它资源

- **getfattr(1)** 手册页面。
- 有关更多信息，请参阅 *Red Hat Ceph Storage 文件系统指南* 中的 *设置文件和目录布局字段* 部分。

8.4. 查看单个布局字段

使用 **getfattr** 命令查看文件或目录的个别布局字段。

先决条件

- 一个正在运行的 Red Hat Ceph Storage 集群。
- 对存储集群中所有节点的根级别访问权限。

流程

- 查看文件或目录中的单个布局字段：

  语法

  ```
  getfattr -n ceph.TYPE.layout.FIELD _PATH
  ```

  替换

  - *TYPE*，**file** 或 **dir**。
  - *FIELD*，字段的名称。
  - 带有到文件或目录路径的 *PATH*。

  示例

  ```
  [root@mon ~]# getfattr -n ceph.file.layout.pool test
  ceph.file.layout.pool="cephfs_data"
  ```

  注意

  pool 字段中的池按照名称来指示。不过，新建的池可以通过 ID 来指示。

其它资源

- **getfattr(1)** 手册页面。

8.5. 删除目录布局

使用 **setfattr** 命令从目录中移除布局。

注意

设置文件布局时，您无法更改或删除该文件。

先决条件

- 具有布局的目录。

流程

1. 从目录中删除布局：

   语法

   ```
   setfattr -x ceph.dir.layout DIRECTORY_PATH
   ```

   示例

   ```
   [user@client ~]$ setfattr -x ceph.dir.layout /home/cephfs
   ```

2. 删除 **pool_namespace** 字段：

   语法

   ```
   setfattr -x ceph.dir.layout.pool_namespace DIRECTORY_PATH
   ```

   示例

   ```
   [user@client ~]$ setfattr -x ceph.dir.layout.pool_namespace /home/cephfs
   ```

   注意

   pool_namespace 字段是唯一可以单独删除的字段。

其它资源

- **setfattr(1)** 手册页

其它资源

- 请参阅 *Red Hat Ceph Storage File System Guide* 中的 *部署 Ceph 文件系统*一节。
- 如需更多信息，请参阅 **getfattr(1)** 手册页。

- 如需更多信息，请参阅 **setfattr(1)** 手册页。

第 9 章 CEPH 文件系统快照

作为存储管理员，您可以获取 Ceph 文件系统(CephFS)目录的时间点快照。CephFS 快照是异步的，您可以选择创建哪个目录快照。

先决条件

- 一个运行良好、健康的 Red Hat Ceph Storage 集群。
- 部署 Ceph 文件系统.

9.1. CEPH 文件系统快照

默认情况下，Ceph 文件系统(CephFS)快照功能在新的 Ceph 文件系统上启用，但必须在现有的 Ceph 文件系统上手动启用。CephFS 快照为 Ceph 文件系统创建不可变的时间点视图。CephFS 快照是异步的，并保存在 CephFS 目录中一个名为 **.snap** 的特殊隐藏目录中。您可以为 Ceph 文件系统中的任何目录指定快照创建。在指定目录时，快照还包含其下的所有子目录。

警告

每个 Ceph 元数据服务器(MDS)集群独立分配 snap 标识符。对共享一个池的多个 Ceph 文件系统使用快照会导致快照冲突，并导致缺少文件数据。

其它资源

- 如需更多详细信息，请参阅 *Red Hat Ceph Storage 文件系统指南* 中的*为 Ceph 文件系统创建快照*部分。
- 如需更多详细信息，请参阅 *Red Hat Ceph Storage 文件系统指南* 中的*为 Ceph 文件系统创建快照调度*部分。

9.2. 为 CEPH 文件系统创建快照

您可以通过创建快照来创建 Ceph 文件系统(CephFS)的不可变、时间点视图。

注意

对于新的 Ceph 文件系统，默认启用快照。

先决条件

- 一个运行良好、健康的 Red Hat Ceph Storage 集群。
- 部署 Ceph 文件系统.
- 对 Ceph 元数据服务器(MDS)节点的根级访问。

流程

1. 登录到 Cephadm shell：

   示例

   ```
   [root@mds ~]# cephadm shell
   ```

2. 对于现有的 Ceph 文件系统，启用快照功能：

   语法

   ```
   ceph fs set FILE_SYSTEM_NAME allow_new_snaps true
   ```

   示例

   ```
   [ceph: root@mds ~]# ceph fs set cephfs01 allow_new_snaps true
   ```

3. 在 **.snap** 目录下创建新快照子目录：

   语法

   ```
   mkdir NEW_DIRECTORY_PATH
   ```

   示例

   ```
   [ceph: root@mds ~]# mkdir /.snap/new-snaps
   ```

   本例创建 **new-snaps** 子目录，这将通知 Ceph 元数据服务器(MDS)来启动快照。

   a. 删除快照：

      语法

      ```
      rmdir NEW_DIRECTORY_PATH
      ```

      示例

      ```
      [ceph: root@mds ~]# rmdir /.snap/new-snaps
      ```

      重要

      尝试删除可能包含底层快照的根级快照将失败。

其它资源

- 如需了解更多详细信息，请参阅 *Red Hat Ceph Storage 文件系统指南* 中的 *Ceph 文件系统快照调度*部分。
- 如需了解更多详细信息，请参阅 *Red Hat Ceph Storage 文件系统指南* 中的 *Ceph 文件系统快照*部分。
- 请参阅 *Red Hat Ceph Storage File System Guide* 中的 *部署 Ceph 文件系统*一节。

第 10 章 CEPH 文件系统快照调度

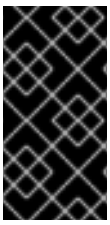
作为存储管理员，您可以获取 Ceph 文件系统(CephFS)目录的时间点快照。CephFS 快照是异步的，您可以选择创建哪个目录快照。

先决条件

- 一个运行良好、健康的 Red Hat Ceph Storage 集群。
- 部署 Ceph 文件系统.

10.1. CEPH 文件系统快照调度

Ceph 文件系统(CephFS)可以调度文件系统目录的快照。快照的调度由 Ceph Manager 管理，它依赖于 Python Timers。快照调度数据作为对象存储在 CephFS 元数据池中，在运行时，所有调度数据都位于序列化 SQLite 数据库中。



重要

调度程序根据指定的时间精确进行精确，以便在存储集群处于正常负载下时使快照保持运行。当 Ceph Manager 处于重度负载时，快照可能不会立即调度，从而导致快照延迟稍有延迟。如果发生这种情况，则下一个调度的快照会象没有延迟的情况下运行。调度的快照会延迟到整个调度中不会导致偏移。

使用方法

Ceph 文件系统(CephFS)调度快照由 **snap_schedule** Ceph Manager 模块管理。此模块提供了一个添加、查询和删除快照调度的接口，以及管理保留策略。此模块还实施 **ceph fs snap-schedule** 命令，其中包含几个子命令来管理计划和保留策略。所有子命令都采用 CephFS 卷路径和子卷路径参数，来指定使用多个 Ceph 文件系统时的文件系统路径。不指定 CephFS 卷路径，参数默认为 **fs_map** 中列出的第一个文件系统，而不指定 subvolume 路径参数，默认为不设置。

快照调度由文件系统路径、重复间隔和开始时间标识。重复间隔定义了两个后续快照之间的时间。间隔格式是一个数字加一个时间设计器：**h**(our), **d**(ay), 或 **w**(eek).例如，间隔为 **4h**，代表每四个小时一个快照。起始时间是一个字符串值，格式为 ISO 格式 **%Y-%m-%dT%H:%M:%S**，如果未指定，则开始时间使用最后午夜的默认值。例如，您可以使用默认的开始时间值将快照调度到 **14:45**，重复间隔为 **1h**，第一个快照将生成 15:00。

保留策略由文件系统路径和保留策略规格来标识。定义保留策略，包括数字加上时间设计器或串联对，格式为 ***COUNT TIME_PERIOD***。策略确保保留了大量快照，并且快照至少在指定的时间段内。时间期限设计器为：**h**(our), **d**(ay), **w**(eek), **m**(onth), **y**(ear), 和 **n**。**n** 时间段设计人员是一个特殊的修饰符，这意味着无论时间如何，保持最后一次快照的数量。例如，**4d** 表示保留至少一天或更长时间的四个快照。

其它资源

- 如需更多详细信息，请参阅 *Red Hat Ceph Storage 文件系统指南* 中的*为 Ceph 文件系统创建快照*部分。
- 如需更多详细信息，请参阅 *Red Hat Ceph Storage 文件系统指南* 中的*为 Ceph 文件系统创建快照调度*部分。

10.2. 为 CEPH 文件系统添加快照调度

为尚不存在的 CephFS 路径添加快照调度。您可以为单个路径创建一个或多个计划。如果计划重复的时间间隔和开始时间不同，则计划会被视为不同的。

CephFS 路径只能具有一个保留策略，但保留策略可以有多个计数期限对。

注意

启用调度程序模块后，运行 **ceph fs snap-schedule** 命令将显示可用的子命令及其用法格式。

先决条件

- 一个运行良好、健康的 Red Hat Ceph Storage 集群。
- 部署 Ceph 文件系统.
- 对 Ceph Manager 和元数据服务器(MDS)节点的 root 级别访问权限。
- 在文件系统中启用 CephFS 快照。

流程

1. 在 Ceph Manager 节点上登录到 Cephadm shell：

   示例

   ```
   [root@host01 ~]# cephadm shell
   ```

2. 启用 **snap_schedule** 模块：

   示例

   ```
   [ceph: root@host01 /]# ceph mgr module enable snap_schedule
   ```

3. 登录到客户端节点：

   示例

   ```
   [root@host02 ~]# cephadm shell
   ```

4. 为 Ceph 文件系统添加新计划：

   语法

   ```
   ceph fs snap-schedule add FILE_SYSTEM_VOLUME_PATH REPEAT_INTERVAL
   [START_TIME] --fs CEPH_FILE_SYSTEM_NAME
   ```

   示例

   ```
   [ceph: root@host02 /]# ceph fs snap-schedule add /cephfs_kernelf739cwtus2/pmo9axbwsi
   1h 2022-06-27T21:50:00 --fs mycephfs
   ```

注意

START_TIME 以 ISO 8601 格式表示。

本例为文件系统 **mycephfs** 的路径 **/cephfs** 创建了一个快照调度，每小时执行一次快照，并在 2022 年 6 月 27 日 9:50 PM 开始。

5. 为 CephFS 卷路径的快照添加新保留策略：

语法

```
ceph fs snap-schedule retention add FILE_SYSTEM_VOLUME_PATH
[COUNT_TIME_PERIOD_PAIR] TIME_PERIOD COUNT
```

示例

```
[ceph: root@host02 /]# ceph fs snap-schedule retention add /cephfs h 14 (1)
[ceph: root@host02 /]# ceph fs snap-schedule retention add /cephfs d 4 (2)
[ceph: root@host02 /]# ceph fs snap-schedule retention add /cephfs 14h4w (3)
```

(1) 这个示例最多保留 14 个快照。

(2) 这个示例只保留 4 个快照。

(3) 这个示例保留 14 个每小时快照和 4 个每周快照。

6. 列出快照计划，以验证新计划是否已创建。

语法

```
ceph fs snap-schedule list FILE_SYSTEM_VOLUME_PATH [--format=plain|json] [--
recursive=true]
```

示例

```
[ceph: root@host02 /]# ceph fs snap-schedule list /cephfs --recursive=true
```

这个示例列出了目录树中的所有计划。

7. 检查快照调度的状态：

语法

```
ceph fs snap-schedule status FILE_SYSTEM_VOLUME_PATH [--format=plain|json]
```

示例

```
[ceph: root@host02 /]# ceph fs snap-schedule status /cephfs --format=json
```

本例以 JSON 格式显示 CephFS **/cephfs** 路径的快照调度状态。如果没有指定默认格式为纯文本。

其它资源

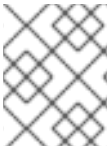
- 如需了解更多详细信息，请参阅 *Red Hat Ceph Storage 文件系统指南* 中的 *Ceph 文件系统快照调度*部分。
- 如需了解更多详细信息，请参阅 *Red Hat Ceph Storage 文件系统指南* 中的 *Ceph 文件系统快照*部分。

10.3. 为 CEPH 文件系统子卷添加快照调度

要管理 Ceph 文件系统(CephFS)子卷快照的保留策略，您可以对单个路径有不同的调度。

如果计划重复的时间间隔和开始时间不同，则计划会被视为不同的。

为尚不存在的 CephFS 文件路径添加快照调度。CephFS 路径只能具有一个保留策略，但保留策略可以有多个计数期限对。



注意

启用调度程序模块后，运行 **ceph fs snap-schedule** 命令将显示可用的子命令及其用法格式。

先决条件

- 部署了 Ceph 文件系统(CephFS)正常工作的 Red Hat Ceph Storage 集群。
- 至少对 Ceph 监控器具有读取访问权限。
- Ceph 管理器节点上的读写功能。
- 已创建 CephFS 子卷和子卷组。

您可以为以下方法创建快照调度：

- 子卷中的目录。
- 默认组中的子卷。
- 非默认组中的子卷。

但是，命令不同。

流程

- 为子卷中的目录创建快照调度：
  a. 获取存在目录的子卷的绝对路径：

  语法

  ```
  ceph fs subvolume getpath VOLUME_NAME SUBVOLUME_NAME
  SUBVOLUME_GROUP_NAME
  ```

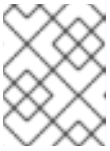
  示例

```
[ceph: root@host02 /]# ceph fs subvolume getpath cephfs subvol_1 subvolgroup_1
```

b. 将快照调度添加到子卷中的目录中：

语法

```
ceph fs snap-schedule add SUBVOLUME_DIR_PATH SNAP_SCHEDULE
[START_TIME] --fs CEPH_FILE_SYSTEM_NAME --subvol SUBVOLUME_NAME
```



注意

snap-schedule 命令中的路径将是 <absolute_path_of_ subvolume>/<relative_path_of_test_dir>，请参阅 子卷的 absolute_path。

示例

```
[ceph: root@host02 /]# ceph fs snap-schedule add
/cephfs_kernelf739cwtus2/pmo9axbwsi 1h 2022-06-27T21:50:00 --fs cephfs --subvol
subvol_1
Schedule set for path /..
```

注意

START_TIME 以 ISO 8601 格式表示。

这个示例为子卷路径创建一个快照调度，每小时快照，并在 2022 年 6 月 27 日 9:50 PM 开始。

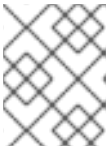
- 要为默认组中的子卷创建快照调度，请运行以下命令：

语法

```
ceph fs snap-schedule add /.. SNAP_SCHEDULE [START_TIME] --fs
CEPH_FILE_SYSTEM_NAME --subvol _SUBVOLUME_NAME
```

示例

```
[ceph: root@host02 /]# ceph fs snap-schedule add - 2M --subvol sv_non_def_1
```



注意

该路径必须定义且无法留空。不依赖于路径字符串值，您可以将其定义为 /, **-'** 或 **/.**。

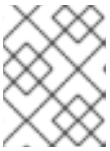
- 要在非默认组中为子卷创建快照调度，请运行以下命令：

语法

```
ceph fs snap-schedule add /.. SNAP_SCHEDULE [START_TIME] --fs
CEPH_FILE_SYSTEM_NAME --subvol _SUBVOLUME_NAME --group
NON_DEFAULT_SUBVOLGROUP_NAME
```

示例

```
[ceph: root@host02 /]# ceph fs snap-schedule add - 2M --fs cephfs --subvol sv_non_def_1 --
group svg1
```



注意

该路径必须定义且无法留空。不依赖于路径字符串值，您可以将其定义为 /, **-'** 或 /.。

10.3.1. 为 CephFS 卷路径的快照调度添加保留策略

要定义任何时间要保留在卷路径中的快照数量，您必须在创建快照调度后添加保留策略。

您可以为子卷组中的目录、默认组中的子卷以及非默认组的子卷创建保留策略。

先决条件

- 一个正在运行的、健康的 IBM Storage Ceph 集群，部署了 Ceph 文件系统(CephFS)。
- 至少对 Ceph 监控器具有读取访问权限。
- Ceph 管理器节点上的读写功能。
- 已创建 CephFS 子卷和子卷组。
- 快照计划。

流程

- 在 CephFS 子卷的目录中为快照调度添加新的保留策略：

  语法

  ```
  ceph fs snap-schedule retention add SUBVOLUME_DIR_PATH
  [COUNT_TIME_PERIOD_PAIR] TIME_PERIOD COUNT
  ```

  示例

  ```
  [ceph: root@host02 /]# ceph fs snap-schedule retention add
  /volumes/_nogroup/subvol_1/85a615da-e8fa-46c1-afc3-0eb8ae64a954/.. h 14 (1)
  [ceph: root@host02 /]# ceph fs snap-schedule retention add
  /volumes/_nogroup/subvol_1/85a615da-e8fa-46c1-afc3-0eb8ae64a954/.. d 4 (2)
  [ceph: root@host02 /]# ceph fs snap-schedule retention add
  /volumes/_nogroup/subvol_1/85a615da-e8fa-46c1-afc3-0eb8ae64a954/.. 14h4w (3)

  Retention added to path /volumes/_nogroup/subvol_1/85a615da-e8fa-46c1-afc3-
  0eb8ae64a954/..
  ```

1. 这个示例最多保留 14 个快照。
2. 这个示例只保留 4 个快照。
3. 这个示例保留 14 个每小时快照和 4 个每周快照。

- 将保留策略添加到默认组中为子卷创建的快照调度中：

  语法

  ```
  ceph fs snap-schedule retention add / [COUNT_TIME_PERIOD_PAIR]
  TIME_PERIOD_COUNT --fs CEPH_FILE_SYSTEM_NAME --subvol SUBVOLUME_NAME
  ```

  示例

  ```
  [ceph: root@host02 /]# ceph fs snap-schedule retention add / 5h --fs cephfs --subvol
  sv_sched
  Retention added to path /volumes/sv_sched/e704342a-ff07-4763-bb0b-a46d9dda6f27/..
  ```

  重要

  必须定义路径(/)，且无法留空。不依赖于路径字符串值，您可以将其定义为 /、- 或 /...

- 将保留策略添加到非默认组中为子卷组创建的快照调度中：

  语法

  ```
  ceph fs snap-schedule retention add / [COUNT_TIME_PERIOD_PAIR]
  TIME_PERIOD_COUNT --fs CEPH_FILE_SYSTEM_NAME --subvol SUBVOLUME_NAME --
  group NON_DEFAULT_SUBVOLGROUP_NAME
  ```

  示例

  ```
  [ceph: root@host02 /]# ceph fs snap-schedule retention add / 5h --fs cephfs --subvol
  sv_sched --group subvolgroup_cg
  Retention added to path /volumes/subvolgroup_cg/sv_sched/e704342a-ff07-4763-bb0b-
  a54j0dda7f16/..
  ```

  重要

  必须定义路径(/)，且无法留空。不依赖于路径字符串值，您可以将其定义为 /、- 或 /...

10.3.2. 列出 CephFS 快照调度

通过列出和遵循快照计划，您可以确保强大的数据保护和高效管理。

先决条件

- 一个正在运行的、健康的 IBM Storage Ceph 集群，部署了 Ceph 文件系统(CephFS)。

- 至少对 Ceph 监控器具有读取访问权限。
- Ceph 管理器节点上的读写功能。
- 已创建 CephFS 子卷和子卷组。
- 快照计划。

流程

- 列出快照计划：

  语法

  ```
  ceph fs snap-schedule list SUBVOLUME_VOLUME_PATH [--format=plain|json] [--recursive=true]
  ```

  示例

  ```
  [ceph: root@host02 /]# ceph fs snap-schedule list / --recursive=true

  /volumes/_nogroup/subv1/85a615da-e8fa-46c1-afc3-0eb8ae64a954/.. 4h
  ```

  这个示例列出了目录树中的所有计划。

10.3.3. 检查 CephFS 快照调度的状态

您可以在子卷的目录中创建的快照中使用 命令检查快照调度的状态，适用于默认子卷组中的子卷，以及非默认组中创建的子卷。

先决条件

- 一个正在运行的、健康的 IBM Storage Ceph 集群，部署了 Ceph 文件系统(CephFS)。
- 至少对 Ceph 监控器具有读取访问权限。
- Ceph 管理器节点上的读写功能。
- 已创建 CephFS 子卷和子卷组。
- 快照计划。

流程

- 检查为子卷中的目录创建的快照调度状态：

  语法

  ```
  ceph fs snap-schedule status SUBVOLUME_DIR_PATH [--format=plain|json]
  ```

  示例

  ```
  [ceph: root@host02 /]# ceph fs snap-schedule status /volumes/_nogroup/subv1/85a615da-
  ```

```
e8fa-46c1-afc3-0eb8ae64a954/.. --format=json

{"fs": "cephfs", "subvol": "subvol_1", "path": "/volumes/_nogroup/subvol_1/85a615da-e8fa-46c1-afc3-0eb8ae64a954/..", "rel_path": "/..", "schedule": "4h", "retention": {"h": 14}, "start": "2022-05-16T14:00:00", "created": "2023-03-20T08:47:18", "first": null, "last": null, "last_pruned": null, "created_count": 0, "pruned_count": 0, "active": true}
```

本例显示 **/volumes/_nogroup/subv1/85a615da-e8fa-46c1-afc3-0eb8ae64a954/..** 路径(JSON 格式)的快照调度状态。如果没有指定默认格式为纯文本。

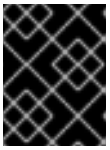
- 检查为 default 组中为子卷创建的快照调度的状态：

  语法

  ```
  ceph fs snap-schedule status --fs CEPH_FILE_SYSTEM_NAME --subvol SUBVOLUME_NAME
  ```

  示例

  ```
  [ceph: root@host02 /]# ceph fs snap-schedule status --fs cephfs --subvol sv_sched
  {"fs": "cephfs", "subvol": "sv_sched", "group": "subvolgroup_cg", "path": "/volumes/subvolgroup_cg/sv_sched/e704342a-ff07-4763-bb0b-a46d9dda6f27/..", "rel_path": "/volumes/subvolgroup_cg/sv_sched/e704342a-ff07-4763-bb0b-a46d9dda6f27/..", "schedule": "1h", "retention": {"h": 5}, "start": "2024-05-21T00:00:00", "created": "2024-05-21T09:18:58", "first": null, "last": null, "last_pruned": null, "created_count": 0, "pruned_count": 0, "active": true}
  ```

  重要

  必须定义路径(/)，且无法留空。不依赖于路径字符串值，您可以将其定义为 /、- 或 /...

- 检查在非默认组中为子卷创建的快照调度状态：.Syntax

```
ceph fs snap-schedule status --fs _CEPH_FILE_SYSTEM_NAME_ --subvol _SUBVOLUME_NAME_ --group _NON-DEFAULT_SUBVOLGROUP_NAME_
```

示例

```
[ceph: root@host02 /]# ceph fs snap-schedule status --fs cephfs --subvol sv_sched --group subvolgroup_cg
{"fs": "cephfs", "subvol": "sv_sched", "group": "subvolgroup_cg", "path": "/volumes/subvolgroup_cg/sv_sched/e564329a-kj87-4763-gh0y-b56c8sev7t23/..", "rel_path": "/volumes/subvolgroup_cg/sv_sched/e704342a-ff07-4763-bb0b-a46d9dda6f27/..", "schedule": "1h", "retention": {"h": 5}, "start": "2024-05-21T00:00:00", "created": "2024-05-21T09:18:58", "first": null, "last": null, "last_pruned": null, "created_count": 0, "pruned_count": 0, "active": true}
```

+ 重要信息：必须定义路径(/)，且无法留空。不依赖于路径字符串值，您可以将其定义为 /、- 或 /...

10.4. 为 CEPH 文件系激活快照调度

本节提供了为 Ceph 文件系统(CephFS)手动将快照调度设置为活动的步骤。

先决条件

- 部署了 Ceph 文件系统(CephFS)正常工作的 Red Hat Ceph Storage 集群。
- 至少对 Ceph 监控器具有读取访问权限。
- Ceph 管理器节点上的读写功能。

流程

- 激活快照调度：

  语法

  ```
  ceph fs snap-schedule activate FILE_SYSTEM_VOLUME_PATH [REPEAT_INTERVAL]
  ```

  示例

  ```
  [ceph: root@host01 /]# ceph fs snap-schedule activate /cephfs
  ```

  这个示例激活 CephFS **/cephfs** 路径的所有调度。

10.5. 为 CEPH 文件系统子卷激活快照调度

本节提供了为 Ceph 文件系统(CephFS)子卷手动将快照调度设置为 active 的步骤。

先决条件

- 部署了 Ceph 文件系统(CephFS)正常工作的 Red Hat Ceph Storage 集群。
- 至少对 Ceph 监控器具有读取访问权限。
- Ceph 管理器节点上的读写功能。

流程

- 激活子卷中为目录创建的快照调度：

  语法

  ```
  ceph fs snap-schedule activate SUBVOL_DIR_PATH [REPEAT_INTERVAL]
  ```

  示例

  ```
  [ceph: root@host01 /]# ceph fs snap-schedule activate
  /volumes/_nogroup/subvol_1/85a615da-e8fa-46c1-afc3-0eb8ae64a954/..
  ```

  本例激活 CephFS **/volumes/_nogroup/subvol_1/85a615da-e8fa-46c1-afc3-0eb8ae64a954/..** 路径的所有调度。

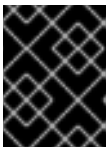
- 激活默认组中为子卷创建的快照调度：

  语法

```
ceph fs snap-schedule activate /.. REPEAT_INTERVAL --fs CEPH_FILE_SYSTEM_NAME --subvol SUBVOLUME_NAME
```

示例

```
[ceph: root@host02 /]# ceph fs snap-schedule activate / --fs cephfs --subvol sv_sched
Schedule activated for path /volumes/subvolgroup_cg/sv_sched/e704342a-ff07-4763-bb0b-a46d9dda6f27/..
```



重要

必须定义路径(/)，且无法留空。不依赖于路径字符串值，您可以将其定义为 /、- 或 /...

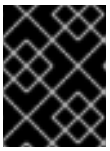
- 激活在非默认组中为子卷创建的快照调度：

语法

```
ceph fs snap-schedule activate /.. [REPEAT_INTERVAL] --fs CEPH_FILE_SYSTEM_NAME --subvol SUBVOLUME_NAME --group NON-DEFAULT_GROUP_NAME
```

示例

```
[ceph: root@host02 /]# ceph fs snap-schedule activate / --fs cephfs --subvol sv_sched --group subvolgroup_cg
Schedule activated for path /volumes/subvolgroup_cg/sv_sched/e704342a-ff07-4763-bb0b-a46d9dda6f27/..
```



重要

必须定义路径(/)，且无法留空。不依赖于路径字符串值，您可以将其定义为 /、- 或 /...

10.6. 为 CEPH 文件系取消激活快照调度

本节提供了为 Ceph 文件系统(CephFS)手动将快照调度设置为 inactive 的步骤。此操作将从调度中排除快照，直到再次激活为止。

先决条件

- 部署了 Ceph 文件系统(CephFS)正常工作的 Red Hat Ceph Storage 集群。
- 至少对 Ceph 监控器具有读取访问权限。
- Ceph 管理器节点上的读写功能。
- 快照计划已创建并处于 active 状态。

流程

- 取消激活 CephFS 路径的快照调度：

语法

```
ceph fs snap-schedule deactivate FILE_SYSTEM_VOLUME_PATH [REPEAT_INTERVAL]
```

示例

```
[ceph: root@host02 ~]# ceph fs snap-schedule deactivate /cephfs 1d
```

本例停用 **/cephfs** 路径的每日快照，从而暂停任何进一步的快照。

10.7. 为 CEPH 文件系统子卷停用快照调度

本节提供了为 Ceph 文件系统(CephFS)子卷手动将快照调度设置为 inactive 的步骤。此操作会将快照从调度中排除，直到再次激活为止。

先决条件

- 部署了 Ceph 文件系统(CephFS)正常工作的 Red Hat Ceph Storage 集群。
- 至少对 Ceph 监控器具有读取访问权限。
- Ceph 管理器节点上的读写功能。
- 快照计划已创建并处于 active 状态。

流程

- 取消激活 CephFS 子卷中的目录的快照调度：

  语法

  ```
  ceph fs snap-schedule deactivate SUBVOL_DIR_PATH [REPEAT_INTERVAL]
  ```

  示例

  ```
  [ceph: root@host02 ~]# ceph fs snap-schedule deactivate
  /volumes/_nogroup/subvol_1/85a615da-e8fa-46c1-afc3-0eb8ae64a954/.. 1d
  ```

  这个示例停用 **/volumes/_nogroup/subvol_1/85a615da-e8fa-46c1-afc3-0eb8ae64a954/..** 路径的每日快照，从而暂停任何进一步的快照。

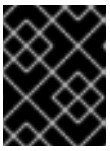
- 取消激活为 default 组中的子卷创建的快照调度：

  语法

  ```
  ceph fs snap-schedule deactivate / REPEAT_INTERVAL --fs CEPH_FILE_SYSTEM_NAME
  --subvol SUBVOLUME_NAME
  ```

  示例

```
[ceph: root@host02 ~]# ceph fs snap-schedule deactivate / --fs cephfs --subvol sv_sched
Schedule deactivated for path /volumes/subvolgroup_cg/sv_sched/e704342a-ff07-4763-
bb0b-a46d9dda6f27/..
```



重要

必须定义路径(/)，且无法留空。不依赖于路径字符串值，您可以将其定义为 /、- 或 /...

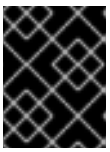
- 取消激活在非默认组中为子卷创建的快照调度：

  语法

  ```
  ceph fs snap-schedule deactivate / REPEAT_INTERVAL --fs CEPH_FILE_SYSTEM_NAME
  --subvol SUBVOLUME_NAME --group NON-DEFAULT_GROUP_NAME
  ```

  示例

  ```
  [ceph: root@host02 ~]# ceph fs snap-schedule deactivate / --fs cephfs --subvol sv_sched --
  group subvolgroup_cg
  Schedule deactivated for path /volumes/subvolgroup_cg/sv_sched/e704342a-ff07-4763-
  bb0b-a46d9dda6f27/..
  ```



重要

必须定义路径(/)，且无法留空。不依赖于路径字符串值，您可以将其定义为 /、- 或 /...

10.8. 删除 CEPH 文件系统的快照调度

本节提供删除 Ceph 文件系统(CephFS)的快照调度的步骤。

先决条件

- 部署了 Ceph 文件系统(CephFS)正常工作的 Red Hat Ceph Storage 集群。
- 至少对 Ceph 监控器具有读取访问权限。
- Ceph 管理器节点上的读写功能。
- 快照计划已创建。

流程

1. 删除特定的快照调度：

   语法

   ```
   ceph fs snap-schedule remove FILE_SYSTEM_VOLUME_PATH [REPEAT_INTERVAL]
   [START_TIME]
   ```

   示例

```
[ceph: root@host02 ~]# ceph fs snap-schedule remove /cephfs 4h 2022-05-16T14:00:00
```

本例删除 **/cephfs** 卷的特定快照调度，该调度每四个小时执行一次，并在 2022 年 5 月 16 日 2:00pm 开始。

2. 删除特定 CephFS 卷路径的所有快照调度：

语法

```
ceph fs snap-schedule remove FILE_SYSTEM_VOLUME_PATH
```

示例

```
[ceph: root@host02 ~]# ceph fs snap-schedule remove /cephfs
```

本例删除 **/cephfs** 卷路径的所有快照调度。

10.9. 删除 CEPH 文件系统子卷的快照调度

本节提供删除 Ceph 文件系统(CephFS)子卷的快照调度步骤。

先决条件

- 部署了 Ceph 文件系统(CephFS)正常工作的 Red Hat Ceph Storage 集群。
- 至少对 Ceph 监控器具有读取访问权限。
- Ceph 管理器节点上的读写功能。
- 快照计划已创建。

流程

- 移除在 CephFS 子卷中为目录创建的特定快照调度：

语法

```
ceph fs snap-schedule remove SUBVOL_DIR_PATH [REPEAT_INTERVAL] [START_TIME]
```

示例

```
[ceph: root@host02 ~]# ceph fs snap-schedule remove
/volumes/_nogroup/subvol_1/85a615da-e8fa-46c1-afc3-0eb8ae64a954/.. 4h 2022-05-
16T14:00:00
```

这个示例删除了 **/volumes/_nogroup/subvol_1/85a615da-e8fa-46c1-afc3-0eb8ae64a954/..** 卷的特定快照调度，每四个小时进行快照，并在 2022 年 5 月 16 日 2:00 PM 开始。

- 删除为 default 组中的子卷创建的特定快照调度：

语法

```
ceph fs snap-schedule remove / --fs CEPH_FILESYSTEM_NAME --subvol
SUBVOLUME_NAME
```

示例

```
[ceph: root@host02 ~]# ceph fs snap-schedule remove / --fs cephfs --subvol sv_sched
Schedule removed for path /volumes/subvolgroup_cg/sv_sched/e704342a-ff07-4763-bb0b-
a46d9dda6f27/..
```

重要

必须定义路径(/)，且无法留空。不依赖于路径字符串值，您可以将其定义为 /、- 或 /...

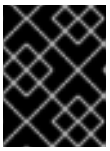
- 删除在非默认组中为子卷创建的特定快照调度：

语法

```
ceph fs snap-schedule remove / --fs CEPH_FILESYSTEM_NAME --subvol
SUBVOLUME_NAME --group NON-DEFAULT_GROUP_NAME
```

示例

```
[ceph: root@host02 ~]# ceph fs snap-schedule remove / --fs cephfs --subvol sv_sched --
group subvolgroup_cg
Schedule removed for path /volumes/subvolgroup_cg/sv_sched/e704342a-ff07-4763-bb0b-
a46d9dda6f27/..
```



重要

必须定义路径(/)，且无法留空。不依赖于路径字符串值，您可以将其定义为 /、- 或 /...

10.10. 删除 CEPH 文件系统的快照调度保留策略

本节提供删除 Ceph 文件系统(CephFS)调度快照的保留策略的步骤。

先决条件

- 部署了 Ceph 文件系统(CephFS)正常工作的 Red Hat Ceph Storage 集群。
- 至少对 Ceph 监控器具有读取访问权限。
- Ceph 管理器节点上的读写功能。
- 为 CephFS 卷路径创建的快照调度。

流程

- 删除 CephFS 路径上的保留策略：

语法

```
ceph fs snap-schedule retention remove FILE_SYSTEM_VOLUME_PATH
[COUNT_TIME_PERIOD_PAIR] TIME_PERIOD COUNT
```

示例

```
[ceph: root@host02 ~]# ceph fs snap-schedule retention remove /cephfs h 4 (1)
[ceph: root@host02 ~]# ceph fs snap-schedule retention remove /cephfs 14d4w (2)
```

1. 这个示例删除 4 个每小时快照。
2. 这个示例删除 14 个每天快照，以及 4 个每周快照。

10.11. 删除 CEPH 文件系统子卷的快照调度保留策略

本节提供删除 Ceph 文件系统(CephFS)子卷的调度快照的保留策略的步骤。

先决条件

- 部署了 Ceph 文件系统(CephFS)正常工作的 Red Hat Ceph Storage 集群。
- 至少对 Ceph 监控器具有读取访问权限。
- Ceph 管理器节点上的读写功能。
- 为 CephFS 子卷路径创建的快照调度。

流程

- 删除 CephFS 子卷中的目录的保留策略：

  语法

  ```
  ceph fs snap-schedule retention remove SUBVOL_DIR_PATH
  [COUNT_TIME_PERIOD_PAIR] TIME_PERIOD COUNT
  ```

  示例

  ```
  [ceph: root@host02 ~]# ceph fs snap-schedule retention remove
  /volumes/_nogroup/subvol_1/85a615da-e8fa-46c1-afc3-0eb8ae64a954/.. h 4 (1)
  [ceph: root@host02 ~]# ceph fs snap-schedule retention remove
  /volumes/_nogroup/subvol_1/85a615da-e8fa-46c1-afc3-0eb8ae64a954/.. 14d4w (2)
  ```

  1. 这个示例删除 4 个每小时快照。
  2. 这个示例删除 14 个每天快照，以及 4 个每周快照。

- 删除在默认组中为子卷的快照调度中创建的保留策略：

  语法

```
ceph fs snap-schedule retention remove / TIME_PERIOD_PAIR TIME_PERIOD COUNT --fs
CEPH_FILESYSTEM_NAME --subvol SUBVOLUME_NAME
```

示例

```
[ceph: root@host02 ~]# ceph fs snap-schedule retention remove / 5h --fs cephfs --subvol
sv_sched --group subvolgroup_cg
Retention removed from path /volumes/subvolgroup_cg/sv_sched/e704342a-ff07-4763-bb0b-
a46d9dda6f27/..
```

重要

必须定义路径(/)，且无法留空。不依赖于路径字符串值，您可以将其定义为 /、- 或 /...

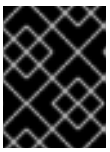
- 删除在非默认组中为子卷的快照调度中创建的保留策略：

语法

```
ceph fs snap-schedule retention remove / TIME_PERIOD_PAIR TIME_PERIOD COUNT --fs
CEPH_FILESYSTEM_NAME --subvol SUBVOLUME_NAME --group NON-
DEFAULT_GROUP_NAME
```

示例

```
[ceph: root@host02 ~]# ceph fs snap-schedule retention remove / 5h --fs cephfs --subvol
sv_sched --group subvolgroup_cg
Retention removed from path /volumes/subvolgroup_cg/sv_sched/e704342a-ff07-4763-bb0b-
a46d9dda6f27/..
```



重要

必须定义路径(/)，且无法留空。不依赖于路径字符串值，您可以将其定义为 /、- 或 /...

- 请参阅 *Red Hat Ceph Storage File System Guide* 中的 *部署 Ceph 文件系统*一节。

第 11 章 CEPH 文件系统快照镜像

作为存储管理员，您可以将 Ceph 文件系统(CephFS)复制到另一个 Red Hat Ceph Storage 集群上的远程 Ceph 文件系统。

先决条件

- 源和目标集群必须运行 Red Hat Ceph Storage 6.0 或更高版本。

Ceph 文件系统(CephFS)支持将快照异步复制到另一个 Red Hat Ceph Storage 集群上的远程 CephFS。快照同步会将快照数据复制到远程 Ceph 文件系统，并在带有相同名称的远程目标上创建新快照。您可以配置特定的目录以进行快照同步。

CephFS 镜像的管理由 CephFS 镜像守护进程(**cephfs-mirror**)进行。此快照数据通过对远程 CephFS 进行批量副本来同步。所选同步快照对的顺序使用 **snap-id** 创建。

重要

不支持同步硬链接。硬链接的文件作为常规文件同步。

CephFS 快照镜像包括功能，如快照调用或高可用性。它们可以通过 Ceph Manager **mirroring** 模块进行管理，这是推荐的控制接口。

Ceph 管理器模块和接口

Ceph Manager **mirroring** 默认为禁用。它提供了用于管理目录快照镜像的接口。Ceph 管理器接口主要是监控用于管理 CephFS 镜像的命令的打包程序。它们是推荐的控制接口。

Ceph Manager **mirroring** 模块是作为 Ceph Manager 插件实现的。它负责分配目录到 **cephfs-mirror** 守护进程以进行同步。

Ceph Manager **mirroring** 模块还提供了一系列命令来控制目录快照的镜像。**mirroring** 模块不管理 **cephfs-mirror** 守护进程。**cephfs-mirror** 守护进程的停止、启动、重新启动和启用由 **systemctl** 控制，但由 **cephadm** 管理。

注意

与 monitor 命令与 **fs mirror** 前缀相比，mirror 模块命令使用 **fs snapshot mirror** 前缀。确保您使用 module 命令前缀来控制目录快照的镜像。

快照警告

可以使用相同名称和不同的内容删除和重新创建快照。用户可以在之前同步"旧的"快照，并在禁用镜像时重新创建快照。使用快照名称推断点的延时会导致"新"快照（警告），永远不会获取同步。

二级文件系统上的快照存储了从中进行同步的快照的 **snap-id**。此元数据存储在 Ceph 元数据服务器上的 **SnapInfo** 结构中。

高可用性

您可以在两个或多个节点上部署多个 **cephfs-mirror** 守护进程，以便在同步目录快照时实现并发。当 **cephfs-mirror** 守护进程部署或终止时，Ceph Manager **mirroring** 模块会发现修改后的 **cephfs-mirror** 守护进程集合，并在新集合之间重新平衡目录分配，从而提供高可用性。

CephFS-mirror 守护进程使用简单的 M/N 策略共享同步负载，其中 M 是目录的数量，N 是 **cephfs-mirror** 守护进程的数量。

重新添加 Ceph 文件系统镜像对等点

在重新添加或重新分配另一个集群中的 CephFS 时，请确保所有镜像守护进程都已停止与对等点同步。您可以使用 **fs mirror status** 命令验证它。Peer UUID 不应显示在命令输出中。

在将同步的目录重新添加到另一个 CephFS 之前，从 peer 中清除同步的目录，特别是新主文件系统中可能存在的目录。如果您要将对等点重新添加到之前从中同步的同一主文件系统，则不需要此项。

其它资源

- 如需有关 **fs mirror status** 命令的更多详细信息，请参阅 *查看 Ceph 文件系统的镜像状态*。

11.1. 为 CEPH 文件系统配置快照镜像

您可以配置 Ceph 文件系统(CephFS)，以镜像(CephFS)在远程 Red Hat Ceph Storage 集群上将快照复制到另一个 CephFS。

注意

同步到远程存储集群所需的时间取决于文件大小和镜像路径中的文件总数。

先决条件

- 源和目标集群必须健康，且运行 Red Hat Ceph Storage 7.0 或更高版本。
- 对源和目标集群中的 Ceph Monitor 节点的 root 级别访问。
- 您的存储集群中至少部署一个 Ceph 文件系统。

流程

1. 登录到 Cephadm shell：

   示例

   ```
   [root@host01 ~]# cephadm shell
   ```

2. 在源存储集群中，部署 CephFS 镜像守护进程：

   语法

   ```
   ceph orch apply cephfs-mirror ["NODE_NAME"]
   ```

   示例

   ```
   [ceph: root@host01 /]# ceph orch apply cephfs-mirror "node1.example.com"
   Scheduled cephfs-mirror update...
   ```

   此命令创建名为 **cephfs-mirror** 的 Ceph 用户，并在给定的节点上部署 **cephfs-mirror** 守护进程。

   a. 可选：部署多个 CephFS 镜像守护进程并实现高可用性：

语法

```
ceph orch apply cephfs-mirror --placement="PLACEMENT_SPECIFICATION"
```

示例

```
[ceph: root@host01 /]# ceph orch apply cephfs-mirror --placement="3 host1 host2 host3"
Scheduled cephfs-mirror update...
```

本例在不同的主机上部署了三个 **cephfs-mirror** 守护进程。

警告

不要将主机与逗号分开，因为它会产生以下错误：

```
Error EINVAL: name component must include only a-z, 0-9, and -
```

3. 在目标存储集群中，为每个 CephFS 对等用户创建一个用户：

语法

```
ceph fs authorize FILE_SYSTEM_NAME CLIENT_NAME / rwps
```

示例

```
[ceph: root@host01 /]# ceph fs authorize cephfs client.mirror_remote / rwps
[client.mirror_remote]
        key = AQCjZ5Jg739AAxAAxduIKoTZbiFJ0lgose8luQ==
```

4. 在源存储集群中，启用 CephFS 镜像模块：

示例

```
[ceph: root@host01 /]# ceph mgr module enable mirroring
```

5. 在源存储集群中，在 Ceph 文件系统上启用镜像功能：

语法

```
ceph fs snapshot mirror enable FILE_SYSTEM_NAME
```

示例

```
[ceph: root@host01 /]# ceph fs snapshot mirror enable cephfs
```

a. 可选：禁用快照镜像：

语法

```
ceph fs snapshot mirror disable FILE_SYSTEM_NAME
```

示例

```
[ceph: root@host01 /]# ceph fs snapshot mirror disable cephfs
```

警告

禁用文件系统上的快照镜像会删除配置的对等点。您需要通过引导方式再次导入同级服务器。

6. 准备目标对等存储集群。

   a. 在目标节点上，启用 **mirroring** Ceph Manager 模块：

   示例

   ```
   [ceph: root@host01 /]# ceph mgr module enable mirroring
   ```

   b. 在同一个目标节点上，创建 peer bootstrap：

   语法

   ```
   ceph fs snapshot mirror peer_bootstrap create FILE_SYSTEM_NAME CLIENT_NAME SITE_NAME
   ```

   SITE_NAME 是用户定义的字符串，用于识别目标存储集群。

   示例

   ```
   [ceph: root@host01 /]# ceph fs snapshot mirror peer_bootstrap create cephfs client.mirror_remote remote-site
   {"token":
   "eyJmc2lkIjogIjBkZjE3MjE3LWRmY2QtNDAzMC05MDc5LTM2Nzk4NTVkNDJlZiIsICJmaWxlc3lzdGVtIjogImJhY2t1cF9mcyIsICJ1c2VyIjogImNsaWVudC5taXJyb3JfcGVlcl9ib290c3RyYXAiLCAic2l0ZV9uYW1lIjogInNpdGUtcmVtb3RlIiwgImtleSI6ICJBUUFhcDBCZ0xtRmpOeEFBVnNyZXozai9YYUV0T2UrbUJEZlJDZz09IiwgIm1vbl9ob3N0IjogIlt2MjoxOTIuMTY4LjAuNTo0MDkxOCx2MToxOTIuMTY4LjAuNTo0MDkxOV0ifQ=="}
   ```

   在双引号之间复制令牌字符串，以便在下一步中使用。

7. 在源存储集群中，从目标存储集群导入 bootstrap 令牌：

   语法

   ```
   ceph fs snapshot mirror peer_bootstrap import FILE_SYSTEM_NAME TOKEN
   ```

-

示例

```
[ceph: root@host01 /]# ceph fs snapshot mirror peer_bootstrap import cephfs
eyJmc2lkIjogIjBkZjE3MjE3LWRmY2QtNDAzMC05MDc5LTM2Nzk4NTVkNDJlZiIsICJmaWxlc3lz
dGVtIjogImJhY2t1cF9mcyIsICJ1c2VyIjogImNsaWVudC5taXJyb3JfcGVlcl9ib290c3RyYXAiLCAic
2l0ZV9uYW1lIjogInNpdGUtcmVtb3RlIiwgImtleSI6ICJBUUFhcDBCZ0xtRmpOeEFBVnNyZXoza
i9YYUV0T2UrbUJEZlJDZz09IiwgIm1vbl9ob3N0IjogIlt2MjoxOTIuMTY4LjAuNTo0MDkxOCx2M
ToxOTIuMTY4LjAuNTo0MDkxOV0ifQ==
```

8. 在源存储集群中，列出 CephFS 镜像对等点：

语法

```
ceph fs snapshot mirror peer_list FILE_SYSTEM_NAME
```

示例

```
[ceph: root@host01 /]# ceph fs snapshot mirror peer_list cephfs
{"e5ecb883-097d-492d-b026-a585d1d7da79": {"client_name": "client.mirror_remote",
"site_name": "remote-site", "fs_name": "cephfs", "mon_host": "
[v2:10.0.211.54:3300/0,v1:10.0.211.54:6789/0]
[v2:10.0.210.56:3300/0,v1:10.0.210.56:6789/0]
[v2:10.0.210.65:3300/0,v1:10.0.210.65:6789/0]"}}
```

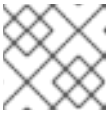
a. 可选：删除快照对等点：

语法

```
ceph fs snapshot mirror peer_remove FILE_SYSTEM_NAME PEER_UUID
```

示例

```
[ceph: root@host01 /]# ceph fs snapshot mirror peer_remove cephfs e5ecb883-097d-
492d-b026-a585d1d7da79
```



注意

请参阅查看 Ceph 文件系统的镜像状态， 了解如何查找对等 UUID 值。

9. 在源存储集群中，为快照镜像配置目录：

语法

```
ceph fs snapshot mirror add FILE_SYSTEM_NAME PATH
```

示例

```
[ceph: root@host01 /]# ceph fs snapshot mirror add cephfs /volumes/_nogroup/subvol_1
```

重要

只有 Ceph 文件系统中的绝对路径有效。

注意

Ceph Manager **mirroring** 模块规范路径。例如，**/d1/d2/./dN** 目录等同于 **/d1/d2**。为镜像添加了目录后，禁止为镜像添加自己的目录和子目录。

a. 可选：停止目录的快照镜像：

语法

```
ceph fs snapshot mirror remove FILE_SYSTEM_NAME PATH
```

示例

```
[ceph: root@host01 /]# ceph fs snapshot mirror remove cephfs /home/user1
```

其它资源

- 如需更多信息，请参阅 *Red Hat Ceph Storage 文件系统指南* 中的*查看 Ceph 文件系统的镜像状态*部分。
- 请参阅 *Red Hat Ceph Storage 文件系统指南* 中的 *Ceph 文件系统镜像*部分。

11.2. 查看 CEPH 文件系统的镜像状态

Ceph 文件系统(CephFS)镜像守护进程(**cephfs-mirror**)会获取 CephFS 镜像状态更改的异步通知，以及对等更新。CephFS 镜像模块提供镜像守护进程状态接口，用于检查镜像守护进程状态。如需更多信息，您可以使用命令查询 **cephfs-mirror** admin socket，以检索镜像状态和对等状态。

先决条件

- 一个正在运行的 Red Hat Ceph Storage 集群。
- 至少部署了一个带有镜像的 Ceph 文件系统。
- 根级别访问运行 CephFS 镜像守护进程的节点。

流程

1. 登录到 Cephadm shell：

   示例

   ```
   [root@host01 ~]# cephadm shell
   ```

2. 检查 **cephfs-mirror** 守护进程状态：

   语法

```
ceph fs snapshot mirror daemon status
```

示例

```
[ceph: root@host01 /]# ceph fs snapshot mirror daemon status
[
  {
    "daemon_id": 15594,
    "filesystems": [
      {
        "filesystem_id": 1,
        "name": "cephfs",
        "directory_count": 1,
        "peers": [
          {
            "uuid": "e5ecb883-097d-492d-b026-a585d1d7da79",
            "remote": {
              "client_name": "client.mirror_remote",
              "cluster_name": "remote-site",
              "fs_name": "cephfs"
            },
            "stats": {
              "failure_count": 1,
              "recovery_count": 0
            }
          }
        ]
      }
    ]
  }
]
```

如需更多信息，请使用 admin socket 接口，如下所示。

3. 在运行 CephFS 镜像守护进程的节点上查找 Ceph 文件系统 ID：

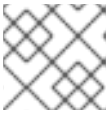
语法

```
ceph --admin-daemon PATH_TO_THE_ASOK_FILE help
```

示例

```
[ceph: root@host01 /]# ceph --admin-daemon /var/run/ceph/1011435c-9e30-4db6-b720-5bf482006e0e/ceph-client.cephfs-mirror.node1.bndvox.asok help
{
    ...
    "fs mirror peer status cephfs@11 1011435c-9e30-4db6-b720-5bf482006e0e": "get peer mirror status",
    "fs mirror status cephfs@11": "get filesystem mirror status",
    ...
}
```

本例中的 Ceph 文件系统 ID 是 **cephfs@11**。



注意

当禁用镜像时，文件系统的对应 **fs mirror status** 命令不会在 **help** 命令中显示。

4. 查看镜像状态：

语法

```
ceph --admin-daemon PATH_TO_THE_ASOK_FILE fs mirror status
FILE_SYSTEM_NAME@_FILE_SYSTEM_ID
```

示例

```
[ceph: root@host01 /]# ceph --admin-daemon /var/run/ceph/1011435c-9e30-4db6-b720-
5bf482006e0e/ceph-client.cephfs-mirror.node1.bndvox.asok fs mirror status cephfs@11
{
  "rados_inst": "192.168.0.5:0/1476644347",
  "peers": {
     "1011435c-9e30-4db6-b720-5bf482006e0e": { (1)
        "remote": {
           "client_name": "client.mirror_remote",
           "cluster_name": "remote-site",
           "fs_name": "cephfs"
        }
     }
  },
  "snap_dirs": {
     "dir_count": 1
  }
}
```

(1) 这是唯一的对等 UUID。

5. 查看对等状态：

语法

```
ceph --admin-daemon PATH_TO_ADMIN_SOCKET fs mirror status
FILE_SYSTEM_NAME@FILE_SYSTEM_ID PEER_UUID
```

示例

```
[ceph: root@host01 /]# ceph --admin-daemon /var/run/ceph/cephfs-mirror.asok fs mirror peer
status cephfs@11 1011435c-9e30-4db6-b720-5bf482006e0e
{
  "/home/user1": {
     "state": "idle", (1)
     "last_synced_snap": {
        "id": 120,
        "name": "snap1",
        "sync_duration": 0.079997898999999997,
        "sync_time_stamp": "274900.558797s"
     },
```

```
    "snaps_synced": 2, 2
    "snaps_deleted": 0, 3
    "snaps_renamed": 0
  }
}
```

state 可以是以下三个值之一：

1. **idle** 表示目录当前没有同步。
2. **syncing** 意味着目录当前正在同步。
3. **failed** 表示目录连续失败已达到上限。

默认连续故障数为 10，默认重试间隔为 60 秒。

1. 显示 **cephfs-mirror** 守护进程映射到的目录：

语法

```
ceph fs snapshot mirror dirmap FILE_SYSTEM_NAME PATH
```

示例

```
[ceph: root@host01 /]# ceph fs snapshot mirror dirmap cephfs /volumes/_nogroup/subvol_1
{
 "instance_id": "25184", 1
 "last_shuffled": 1661162007.012663,
 "state": "mapped"
}
```

1. **instance_id** 是与 **cephfs-mirror** 守护进程关联的 RADOS instance-ID。

示例

```
[ceph: root@host01 /]# ceph fs snapshot mirror dirmap cephfs /volumes/_nogroup/subvol_1
{
 "reason": "no mirror daemons running",
 "state": "stalled" 1
}
```

1. **stalled** 状态表示 CephFS 镜像停滞。

第二个示例显示了没有镜像守护进程运行时的命令输出。

其它资源

- 请参阅 *Red Hat Ceph Storage 文件系统指南* 中的 *Ceph 文件系统镜像*部分。

11.3. 查看 CEPH 文件系统快照镜像的指标

查看这些指标有助于监控性能和同步进度。使用计数器转储检查 Ceph 文件系统快照镜像健康和卷指标。

先决条件

- 正在运行的 IBM Storage Ceph 集群。
- 至少启用了 Ceph 文件系统快照镜像的一个部署。
- 对运行 Ceph 文件系统镜像守护进程的节点的根级别访问权限。

流程

1. 获取 **asok** 文件的名称。**asok** 文件可用，镜像守护进程正在运行，并位于 cephadm shell 中的 **/var/run/ceph/** 中。
2. 在运行 CephFS 镜像守护进程的节点上运行以下命令来检查镜像指标和同步状态。

   语法

   ```
   [ceph: root@mirror-host01 /]# ceph --admin-daemon ASOK_FILE_NAME counter dump
   ```

   示例

   ```
   [ceph: root@mirror-host01 /]# ceph --admin-daemon ceph-client.cephfs-mirror.ceph1-hk-n-0mfqao-node7.pnbrlu.2.93909288073464.asok counter dump

   [
    {
     "key": "cephfs_mirror",
     "value": [
      {
       "labels": {},
       "counters": {
        "mirrored_filesystems": 1,
        "mirror_enable_failures": 0
       }
      }
     ]
    },
    {
     "key": "cephfs_mirror_mirrored_filesystems",
     "value": [
      {
       "labels": {
        "filesystem": "cephfs"
       },
       "counters": {
        "mirroring_peers": 1,
        "directory_count": 1
       }
      }
     ]
    },
    {
     "key": "cephfs_mirror_peers",
   ```

```
    "value": [
     {
      "labels": {
       "peer_cluster_filesystem": "cephfs",
       "peer_cluster_name": "remote_site",
       "source_filesystem": "cephfs",
       "source_fscid": "1"
      },
      "counters": {
       "snaps_synced": 1,
       "snaps_deleted": 0,
       "snaps_renamed": 0,
       "sync_failures": 0,
       "avg_sync_time": {
        "avgcount": 1,
        "sum": 4.216959457,
        "avgtime": 4.216959457
       },
       "sync_bytes": 132
      }
     }
    ]
  }
]
```


指标描述：

标记 **Perf** Counters 生成指标，这些指标可由 OCP/ODF 仪表板使用，以便在 OCP 和 ACM 仪表板和其它地方提供地理复制。

这将生成 cephfs_mirror 同步的进度，并提供监控功能。导出的指标根据以下警报启用监控：

mirroring_peers

镜像涉及的对等点数量。

directory_count

正在同步的目录总数。

mirrored_filesystems

要镜像的文件系统总数。

mirror_enable_failures

启用镜像失败。

snaps_synced

成功同步快照总数。

sync_bytes

正在同步的总字节

sync_failures

失败的快照同步总数。

snaps_deleted

已删除的快照总数。

snaps_renamed

重命名的快照总数。

avg_synced_time

所有快照同步的平均时间。

last_synced_start

最后一次同步快照的同步开始时间。

last_synced_end

最后一次同步快照的同步结束时间。

last_synced_duration

最后一次同步的持续时间。

last_synced_bytes

为最后一个同步快照同步的总字节。

其它资源

- 请参阅 *Red Hat Ceph Storage File System Guide* 中的*部署 Ceph 文件系统*一节。
- 详情请参阅 *Red Hat Ceph Storage 安装指南*。
- 详情请参阅 *Red Hat Ceph Storage 文件系统指南* 中的 *Ceph 文件系统元数据服务器*部分。
- 如需了解更多详细信息，请参阅 *Red Hat Ceph Storage 文件系统指南* 中的 *Ceph 文件系统镜像*部分。

附录 A. CEPH 文件系统的健康状况消息

集群健康检查

Ceph 监控守护进程生成健康消息，以响应元数据服务器(MDS)的某些状态。以下是健康信息及其解释列表：

MDS 排名 <ranks> 失败

目前没有将一个或多个 MDS 等级分配给任何 MDS 守护进程。在合适的替代守护进程启动后，存储集群才会恢复。

MDS 排名 <ranks> 已损坏

一个或多个 MDS 级别对其存储元数据造成严重损坏，在修复元数据之前，无法重新开始。

MDS 集群被降级

当前没有启动和运行 MDS 等级，客户端可能会暂停元数据 I/O，直到此情况解决为止。这包括等级失败或损坏，包括 MDS 上运行但未处于**活动**状态的等级，例如，排名在**重播**状态。

MDS <names> 滞后

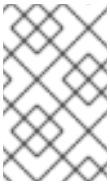
MDS 守护进程应当会以 **mds_beacon_interval** 选项指定的间隔向 monitor 发送 beacon 消息，默认为 4 秒。如果 MDS 守护进程无法在它们 **mds_beacon_grace** 选项指定的时间内发送消息，则默认为 15 秒。Ceph 监控器将 MDS 守护进程标记为 **滞后**，并自动将其替换为待机守护进程（如果有可用）。

守护进程报告的健康检查

MDS 守护进程可以识别各种不需要的条件，并在 **ceph status** 命令的输出中返回它们。这些条件具有人类可读的消息，也具有唯一代码，从 **MDS_HEALTH** 开始，显示在 JSON 输出中。下面列出了守护进程消息、它们的代码和说明。

"Behind on trimming..."

Code: MDS_HEALTH_TRIM
CephFS 维护一个划分为日志段的元数据日志。日志的长度（按分段数）由 **mds_log_max_segments** 设置控制。当片段数超过该设置时，MDS 开始写回元数据，以便可以删除 (trim) 最旧的片段。如果这个过程太慢，或者软件错误正在防止修剪，则会出现这个健康信息。出现此消息的阈值是将网段数量为 **mds_log_max_segments** 的两倍。



注意

如果遇到修剪警告，则建议增加 **mds_log_max_segments**。但是，确保在集群运行状况恢复时将此配置重新重置为默认值，并且不再看到 trim 警告。建议将 **mds_log_max_segments** 设置为 256，以允许 MDS 使用修剪来捕获。

"Client <name> failing to respond to capability release"

Code: MDS_HEALTH_CLIENT_LATE_RELEASE, MDS_HEALTH_CLIENT_LATE_RELEASE_MANY
CephFS 客户端由 MDS 发布。这些能力像锁定一样工作。有时，例如当另一个客户端需要访问权限时，MDS 会请求客户端释放其功能。如果客户端不响应，它可能无法立即这样做，或者根本无法这样做。如果客户端需要的时间超过其 **mds_revoke_cap_timeout** 选项（默认为 60 秒）所指定的时间，将显示此消息。

"Client <name> failing to respond to cache pressure"

Code: MDS_HEALTH_CLIENT_RECALL, MDS_HEALTH_CLIENT_RECALL_MANY
客户端维护元数据缓存。客户端缓存中的项目（如索引节点）也固定在 MDS 缓存中。当 MDS 需要缩小其缓存以保持在自己的缓存大小限制内时，MDS 会将消息发送到客户端，以缩小其缓存。如果客户

端不响应，它可以阻止 MDS 正确保持其缓存大小，并且 MDS 最终可能会耗尽内存并意外终止。如果客户端需要的时间超过其 **mds_recall_state_timeout** 选项（默认为 60 秒）所指定的时间，将显示此消息。详情请参阅元数据服务器缓存大小限制部分。

"Client name failing to advance its oldest client/flush tid"

Code: MDS_HEALTH_CLIENT_OLDEST_TID, MDS_HEALTH_CLIENT_OLDEST_TID_MANY
用于客户端和 MDS 服务器之间通信的 CephFS 协议使用名为 **oldest tid** 的字段来通知 MDS 完全完成哪些客户端请求，以便 MDS 可以忘记它们。如果一个不响应的客户端未能推进此字段，可能会阻止 MDS 正确清理客户端请求使用的资源。如果客户端的请求数超过 **max_completed_requests** 选项（默认为 100000）指定的数量（默认为 100000），则会出现在 MDS 侧完成但还没有在客户端 **最旧的 tid** 值中考虑的请求数。

"Metadata damage detected"

Code: MDS_HEALTH_DAMAGE
从元数据池中读取时会出现元数据损坏或缺失的问题。此消息表示损坏已完全隔离，使 MDS 能够继续运行，尽管客户端访问损坏的子树会返回 I/O 错误。使用 **damage ls** 管理 socket 命令查看损坏的详细信息。一旦遇到损坏，就会发出此消息。

"MDS in read-only mode"

Code: MDS_HEALTH_READ_ONLY
MDS 已进入只读模式，并将 **EROFS** 错误代码返回尝试修改任何元数据的客户端操作。MDS 进入只读模式：

- 如果在写入元数据池时遇到写入错误。
- 如果管理员强制 MDS 使用 **force_readonly** 管理 socket 命令进入只读模式。

"<N> slow requests are blocked"

Code: MDS_HEALTH_SLOW_REQUEST
一个或多个客户端请求尚未立即完成，这表示 MDS 运行缓慢或遇到漏洞。使用 **ops** 管理 socket 命令列出出色的元数据操作。如果任何客户端请求的时间超过其 **mds_op_complaint_time** 选项（默认为 30 秒）指定的值，则会出现此消息。

"Too many inodes in cache"

Code: MDS_HEALTH_CACHE_OVERSIZED

MDS 无法修剪其缓存，以遵守管理员设置的限制。如果 MDS 缓存太大，守护进程可能会耗尽可用内存并意外终止。默认情况下，如果 MDS 缓存大小大于其限制的 50%，则会出现此消息。

其它资源

- 详情请参阅 *Red Hat Ceph Storage 文件系统指南中的 元数据服务器缓存大小限制* 部分。

附录 B. 元数据服务器守护进程配置参考

请参阅此列出可用于元数据服务器(MDS)守护进程配置的命令列表。

mon_force_standby_active

描述

如果设置为 **true**，请监控在待机重播模式中强制 MDS 处于活动状态。在 Ceph 配置文件的 **[mon]** 或 **[global]** 部分下设置。

类型

布尔值

默认

true

max_mds

描述

集群创建过程中活跃 MDS 守护进程的数量。在 Ceph 配置文件的 **[mon]** 或 **[global]** 部分下设置。

类型

32 位整数

默认

1

mds_cache_memory_limit

描述

内存限制 MDS 为其缓存强制执行的 MDS。红帽建议使用这个参数而不是它们 **mds cache size** 参数。

类型

64 位 Unsigned 整数

默认

4294967296

mds_cache_reservation

描述

维护 MDS 缓存的缓存保留、内存或索引节点。该值是配置的最大缓存的百分比。MDS 开始转换为保留时，它会重新调用客户端状态，直到其缓存大小缩小来恢复保留为止。

类型

浮点值

默认

0.05

mds_cache_size

描述

要缓存的索引节点数。值 0 表示无限数字。红帽建议使用它们 **mds_cache_memory_limit** 来限制 MDS 缓存使用的内存量。

类型

32 位整数

默认

0

mds_cache_mid

描述

从顶部的缓存 LRU 中新项目的插入点。

类型

浮点值

默认

0.7

mds_dir_commit_ratio

描述

部分的目录包含 Ceph 提交之前使用完整更新（而非部分更新）的错误信息。

类型

浮点值

默认

0.5

mds_dir_max_commit_size

描述

Ceph 将目录更新前的最大目录大小（以 MB 为单位）将目录分成较小的事务。

类型

32 位整数

默认

90

mds_decay_halflife

描述

MDS 缓存温度的半寿命。

类型

浮点值

默认

5

mds_beacon_interval

描述

发送到 monitor 的 beacon 消息的频率（以秒为单位）。

类型

浮点值

默认

4

mds_beacon_grace

描述

Ceph 声明 MDS **滞后**的时间间隔，并且可能替换它。

类型

浮点值

默认

15

mds_blacklist_interval

描述

OSD map 中失败 MDS 守护进程的黑名单持续时间。

类型

浮点值

默认

24.0*60.0

mds_session_timeout

描述

Ceph 超出功能和租用前客户端不活跃的时间间隔，以秒为单位。

类型

浮点值

默认

60

mds_session_autoclose

描述

Ceph 关闭**滞后**客户端会话前的时间间隔（以秒为单位）。

类型

浮点值

默认

300

mds_reconnect_timeout

描述

在 MDS 重启期间等待客户端重新连接的时间间隔（以秒为单位）。

类型

浮点值

默认

45

mds_tick_interval

描述

MDS 执行内部定期任务的频率。

类型

浮点值

默认

5

mds_dirstat_min_interval

描述

尝试将递归统计信息传播到树上的最小间隔（以秒为单位）。

类型

浮点值

默认

1

mds_scatter_nudge_interval

描述

目录统计数据更改的速度传播。

类型

浮点值

默认

5

mds_client_prealloc_inos

描述

每个客户端会话预分配的索引节点编号.

类型

32 位整数

默认

1000

mds_early_reply

描述

确定 MDS 是否允许客户端在提交到日志之前查看请求结果。

类型

布尔值

默认

true

mds_use_tmap

描述

使用 **trivialmap** 进行目录更新。

类型

布尔值

默认

true

mds_default_dir_hash

描述

用于跨目录片段的散列文件的功能。

类型

32 位整数

默认

2，是 rjenkins

mds_log

描述

如果 MDS 应记录元数据更新，则设置为 **true**。禁用仅进行基准测试。

类型

布尔值

默认

true

mds_log_skip_corrupt_events

描述

决定 MDS 是否尝试在日志重播期间跳过损坏的日志事件。

类型

布尔值

默认

false

mds_log_max_events

描述

Ceph 启动修剪前日志中的最大事件。设置为 **-1** 以禁用限制。

类型

32 位整数

默认

-1

mds_log_max_segments

描述

Ceph 启动修剪之前的日志中的最大网段或对象数量。设置为 **-1** 以禁用限制。

类型

32 位整数

默认

30

mds_log_max_expiring

描述

并行过期的最大片段数。

类型

32 位整数

默认

20

mds_log_eopen_size

描述

EOpen 事件中索引节点的最大数量.

类型

32 位整数

默认

100

mds_bal_sample_interval

描述

决定在做出碎片决策时，对目录温度进行抽样的频率。

类型

浮点值

默认

3

mds_bal_replicate_threshold

描述

Ceph 尝试将元数据复制到其他节点前的最大温度。

类型

浮点值

默认

8000

mds_bal_unreplicate_threshold

描述

Ceph 停止将元数据复制到其他节点前的最小温度。

类型

浮点值

默认

0

mds_bal_frag

描述

决定是否 MDS 分片目录。

类型

布尔值

默认

false

mds_bal_split_size

描述

MDS 将目录片段分割为更小的位前的最大目录大小。根目录的默认片段大小限制为 10000。

类型

32 位整数

默认

10000

mds_bal_split_rd

描述

Ceph 分割目录片段之前的最大目录读取温度。

类型

浮点值

默认

25000

mds_bal_split_wr

描述

Ceph 分割目录片段之前的最大目录写入温度。

类型

浮点值

默认

10000

mds_bal_split_bits

描述

用于分割目录片段的位数。

类型

32 位整数

默认

3

mds_bal_merge_size

描述

Ceph 尝试合并相邻目录片段前的最小目录大小。

类型

32 位整数

默认

50

mds_bal_merge_rd

描述

Ceph 合并相邻目录片段之前读取的最小温度。

类型

浮点值

默认

1000

mds_bal_merge_wr

描述

Ceph 合并相邻目录片段之前，最小写入温度。

类型

浮点值

默认

1000

mds_bal_interval

描述

MDS 节点之间工作负载交换的频率（以秒为单位）。

类型

32 位整数

默认

10

mds_bal_fragment_interval

描述

调整目录碎片的频率（以秒为单位）。

类型

32 位整数

默认

5

mds_bal_idle_threshold

描述

Ceph 将子树迁移到其父树前的最小温度。

类型

浮点值

默认

0

mds_bal_max

描述

Ceph 停止前要运行均衡器的迭代数量。仅用于测试目的。

类型

32 位整数

默认

-1

mds_bal_max_until

描述

Ceph 停止前运行平衡的秒数。仅用于测试目的。

类型

32 位整数

默认

-1

mds_bal_mode

描述

计算 MDS 负载的方法：

- **1** = 混合。
- **2** = 请求率和延迟。
- **3** = CPU 负载。

类型

32 位整数

默认

0

mds_bal_min_rebalance

描述

Ceph 迁移前的最小子树温度。

类型

浮点值

默认

0.1

mds_bal_min_start

描述

Ceph 搜索子树前的最小子树温度。

类型

浮点值

默认

0.2

mds_bal_need_min

描述

要接受的目标子树大小的最小部分。

类型

浮点值

默认

0.8

mds_bal_need_max

描述

要接受的目标子树大小的最大部分。

类型

浮点值

默认

1.2

mds_bal_midchunk

描述

Ceph 迁移任何大于目标子树大小的子树。

类型

浮点值

默认

0.3

mds_bal_minchunk

描述

Ceph 会忽略小于目标子树大小的这一部分的任何子树。

类型

浮点值

默认

0.001

mds_bal_target_removal_min

描述

Ceph 从 MDS 映射中删除旧 MDS 目标前的最小负载均衡器迭代数量。

类型

32 位整数

默认

5

mds_bal_target_removal_max

描述

Ceph 从 MDS map 移除旧 MDS 目标前的最大均衡迭代数量。

类型

32 位整数

默认

10

mds_replay_interval

描述

当 **hot standby** 处于 **standby-replay** 模式时，日志的拉取间隔。

类型

浮点值

默认

1

mds_shutdown_check

描述

MDS 关闭期间轮询缓存的时间间隔。

类型

32 位整数

默认

0

mds_thrash_exports

描述

Ceph 在节点之间随机导出子树。仅用于测试目的。

类型

32 位整数

默认

0

mds_thrash_fragments

描述

Ceph 随机分割或合并目录。

类型

32 位整数

默认

0

mds_dump_cache_on_map

描述

Ceph 将 MDS 缓存内容转储到每个 MDS 映射上的一个文件。

类型

布尔值

默认

false

mds_dump_cache_after_rejoin

描述

Ceph 在恢复期间重新加入缓存后，将 MDS 缓存内容转储到文件。

类型

布尔值

默认

false

mds_verify_scatter

描述

Ceph 声称各种 scatter/gather invariants 是 **true**。仅供开发人员使用。

类型

布尔值

默认

false

mds_debug_scatterstat

描述

Ceph 声称各种递归统计在变量中是 **true**。仅供开发人员使用。

类型

布尔值

默认

false

mds_debug_frag

描述

Ceph 在方便时验证目录碎片不变量。仅供开发人员使用。

类型

布尔值

默认

false

mds_debug_auth_pins

描述

debug 身份验证固定变量。仅供开发人员使用。

类型

布尔值

默认

false

mds_debug_subtrees

描述

调试子树变量.仅供开发人员使用。

类型

布尔值

默认

false

mds_kill_mdstable_at

描述

Ceph 在 MDS 表代码中注入 MDS 故障。仅供开发人员使用。

类型

32 位整数

默认

0

mds_kill_export_at

描述

Ceph 在子树导出代码中注入 MDS 失败。仅供开发人员使用。

类型

32 位整数

默认

0

mds_kill_import_at

描述

Ceph 在子树导入代码中注入 MDS 失败。仅供开发人员使用。

类型

32 位整数

默认

0

mds_kill_link_at

描述

Ceph 在硬链接代码中注入 MDS 故障。仅供开发人员使用。

类型

32 位整数

默认

0

mds_kill_rename_at

描述

Ceph 在重命名代码中注入 MDS 失败。仅供开发人员使用。

类型

32 位整数

默认

0

mds_wipe_sessions

描述

Ceph 会在启动时删除所有客户端会话。仅用于测试目的。

类型

布尔值

默认

0

mds_wipe_ino_prealloc

描述

Ceph 在启动时删除内节点预分配元数据。仅用于测试目的。

类型

布尔值

默认

0

mds_skip_ino

描述

启动时要跳过的索引节点编号数。仅用于测试目的。

类型

32 位整数

默认

0

mds_standby_for_name

描述

MDS 守护进程对此设置中指定的名称的另一个 MDS 守护进程是备用的。

类型

字符串

默认

N/A

mds_standby_for_rank

描述

MDS 守护进程的实例是这个等级的另一个 MDS 守护进程实例的备用实例。

类型

32 位整数

默认

-1

mds_standby_replay

描述

确定 MDS 守护进程是否在用作 **hot standby** 时轮询和重播活动 MDS 的日志。

类型

布尔值

默认

false

附录 C. JOURNALER 配置参考

可用于日志记录器配置的列表命令的引用。

journaler_write_head_interval

描述

更新 journal head 对象的频率。

类型

整数

必需

否

默认

15

journaler_prefetch_periods

描述

在日志重播上提前读取多少条带周期.

类型

整数

必需

否

默认

10

journaler_prezero_periods

描述

写入位置前要达到零的条带周期数。

类型

整数

必需

否

默认

10

journaler_batch_interval

描述

人为地发生最大额外延迟（以秒为单位）。

类型

双

必需

否

默认

.001

journaler_batch_max

描述

将延迟刷新的最大字节数。

类型

64-bit Unsigned 整数

必需

否

默认

0

附录 D. CEPH 文件系统镜像配置参考

本节列出了 Ceph 文件系统(CephFS)镜像的配置选项。

cephfs_mirror_max_concurrent_directory_syncs

描述

cephfs-mirror 守护进程可以同时同步的目录快照的最大数量。控制同步线程的数量。

类型

整数

默认

3

Min

1

cephfs_mirror_action_update_interval

描述

处理待处理的镜像更新操作的间隔（以秒为单位）。

类型

secs

默认

2

Min

1

cephfs_mirror_restart_mirror_on_blocklist_interval

描述

重启阻塞镜像实例的间隔（以秒为单位）。设置为零(**0**)可禁用重启阻止的实例。

类型

secs

默认

30

Min

0

cephfs_mirror_max_snapshot_sync_per_cycle

描述

当一个目录被 worker 线程获取镜像时，要镜像的最大快照数量。

类型

整数

默认

3

Min

1

cephfs_mirror_directory_scan_interval

描述

为快照镜像扫描配置的目录的时间间隔（以秒为单位）。

类型

整数

默认

10

Min

1

cephfs_mirror_max_consecutive_failures_per_directory

描述

将目录标记为"failed"的连续快照同步失败的数量。失败的目录会重试，以便更频繁地进行同步。

类型

整数

默认

10

Min

0

cephfs_mirror_retry_failed_directories_interval

描述

对失败目录重试同步的时间间隔（以秒为单位）。

类型

整数

默认

60

Min

1

cephfs_mirror_restart_mirror_on_failure_interval

描述

重启失败的镜像实例的间隔（以秒为单位）。设置为零(**0**)可禁用重启失败的镜像实例。

类型

secs

默认

20

Min

0

cephfs_mirror_mount_timeout

描述

由 **cephfs-mirror** 守护进程挂载主 CephFS 或辅助 CephFS 的超时时间（以秒为单位）。把它设置为更高的值可能会导致在挂载文件系统时镜像守护进程停滞（如果集群无法访问）。此选项用于覆盖常见的 **client_mount_timeout**。

类型

secs

默认

10

Min

0